1993-03

Breakdown of adiabatic invariance

Leigh, Charlotte V.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

BREAKDOWN OF ADIABATIC INVARIANCE

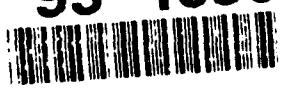
by

Charlotte V. Leigh

March 1993

Thesis Advisor: Bruce Denardo

Approved for public release; distribution is unlimited



Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REFORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b. OFFICE SYMBOL (If applicable) PH	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Breakdown of Adiabatic Invariance

12. PERSONAL AUTHOR(S)
Charlotte V. Leigh

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year,Month,Day)	15. PAGE COUNT
Master's Thesis	FROM TO	March 1993	84

16. SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Adiabatic Invariance

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Adiabatic invariance, in which certain quantities of a system remain unchanged as a parameter of the system is infinitely slowly altered, plays a fundamental role in many areas of physics. For any harmonic oscillator, the adiabatic invariant is the energy divided by the frequency. When the alterations are slow but occur over a finite time, there is predicted to be an *exponential suppression* of the change in adiabtic invariant; that is, if ε is a dimensionless positive number that tends to zero in the limit of infinitely slow alterations, then the change in adiabatic invariant is proportional to $\exp(-1/\varepsilon)$. We report numerical simulations of three oscillators whose parameters are varied at rates ranging from very slow to very fast compared to the oscillation frequency. The models are single-degree-of-freedom oscillators that are based on simple physical systems. The exponential suppression is not observed, which indicates that its observation may be extremely difficult or impossible. Furthermore, the change in adiabatic invariant is found to depend upon the initial phase even in the limit of infinitely slow changes. In the case of abrupt alterations, the numerical simulations verify some theoretical calculations, but reveal that other theoretical calculations are incorrect.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
[X] UNCLASSIFIED/UNLIMITED [ ] SAME AS RPT. [ ] DTIC USERS	Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Bruce C. Denardo	22b. TELEPHONE (Include Area Code) 408-646-3485 22c. OFFICE SYMBOL PH/De

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603 Unclassified

Approved for public release; distribution is unlimited

Breakdown of Adiabatic Invariance

by

Charlotte V. Leigh

Lieutenant Commander, United States Navy
B.S., Hawaii Pacific University, 1991

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING ACOUSTICS

from the

NAVAL POSTGRADUATE SCHOOL

March, 1993

Author: [redacted]
Charlotte V. Leigh

Approved by: [redacted]
Bruce Denardo, Thesis Advisor

[redacted]
Andres Larraza, Second Reader

[redacted]
Anthony A. Atchley, Chairman,
Engineering Acoustics Academic Committee

ABSTRACT

Adiabatic invariance, in which certain quantities of a system remain unchanged as a parameter of the system is infinitely slowly altered, plays a fundamental role in many areas of physics. For any harmonic oscillator, the adiabatic invariant is the energy divided by the frequency. When the alterations are slow but occur over a finite time, there is predicted to be an *exponential suppression* of the change in adiabtic invariant; that is, if ε is a dimensionless positive number that tends to zero in the limit of infinitely slow alterations, then the change in adiabatic invariant is proportional to $\exp(-1/\varepsilon)$. We report numerical simulations of three oscillators whose parameters are varied at rates ranging from very slow to very fast compared to the oscillation frequency. The models are single-degree-of-freedom oscillators that are based on simple physical systems. The exponential suppression is not observed, which indicates that its observation may be extremely difficult or impossible. Furthermore, the change in adiabatic invariant is found to depend upon the initial phase even in the limit of infinitely slow changes. In the case of abrupt alterations, the numerical simulations verify some theoretical calculations, but reveal that other theoretical calculations are incorrect.

TABLE OF CONTENTS

I. INTRODUCTION

Conserved quantities (constants of the motion) of a system are important both fundamentally and practically. In any closed system, for example, the total energy is constant. In open systems, no conserved quantities exist in general. However, for cases in which an external agent slowly alters a parameter of a system, approximate constants do exist. These are referred to as adiabatic invariants, and are exactly conserved in the limit of infinitely slow alterations.

Adiabatic invariance plays a fundamental role in much of physics. For example, the entropy of a closed thermodynamic system remains constant as the system is subjected to slow mechanical changes. The underlying quantum statistical description of this process is that the occupation numbers corresponding to the energy eigenstates are constant during such changes. In fact, the "old" quantum theory was based upon the axiom of quantifying the classical adiabatic invariants of the motion. Many practical techniques, including geometric ray theory in underwater acoustics and the bending of microwave radiation in waveguides, owe both their utility and limitations to adiabatic invariance. The breakdown of adiabatic invariance in the latter example is the loss of energy of the primary mode as a result of the excitation of other propagating modes, including those in the backward direction. Because microwaveguides typically operate in frequency ranges such that there is only one type of mode propagating, the new excitation is limited to the backward-propagating mode. Adiabatic invariance has also been employed in a variety of simple systems (Kubo, 1967; Crawford, 1990; Denardo and Alkov, 1993).

When external changes are infinitely smooth but not infinitely slow, a weak

breakdown of adiabatic invariance is predicted. The change in the adiabatic invariant of the initially excited state, and the excitation of other states, are exponentially suppressed. By this is meant that, if ε is a dimensionless positive number which tends to zero in the limit of infinitely slow external changes, then the changes in the adiabatic invariants of the system are proportional to $\exp(-1/\varepsilon)$ (Landau and Lifshitz, 1976; Corben and Stehle, 1960). This effect has recently been considered in quantum systems (Kvitsinsky and Putterman, 1990). For changes that are not infinitely smooth, the suppression is predicted to be algebraic rather than exponential; the changes are proportional to ε^n, where n is the order of the derivative that is discontinuous (Lenard, 1959). Because no external changes in real or computational systems can be infinitely smooth, observation of the exponential suppression is not expected. The nature of the breakdown may thus serve as a probe of the smoothness of the external changes. To our knowledge, the breakdown of adiabatic invariance has not been quantified experimentally or numerically.

In this thesis, we employ numerical simulations to examine the breakdown of adiabatic invariance in simple oscillators with one degree of freedom. Three such model systems are investigated: a longitudinal mass-and-spring oscillator confined to a tube that is rotated 360° about a perpendicular bisector of the tube, the same system but translated one unit distance along the axis of the tube, and a transverse mass-and-spring oscillator where the walls connected to the springs are both moved either outward or inward. We refer to the first as the *Rotate* system, the second as the *Translate* system, and the third as the *Dilate* system.

In Ch. II, we describe out three model systems and theoretically consider adiabatic invariance and its breakdown. The adiabatic invariance of one of the systems is explicitly derived. We also derive the final states of the systems in the limit of abrupt alterations; i.e., when the alterations occur over a time that is much smaller than the characteristic

period of the oscillations. In Ch. III, we describe our numerical method and present results for the three systems. Comparison with theory is made in both the near-adiabatic (slow but not infinitely slow) regime and the abrupt regime. In Ch. IV, we present conclusions and describe a substantial amount of future work. The three appendices consist of the C-language computer programs that we created and used.

II. THEORY

A. MODEL SYSTEMS

In this section, we derive the equations of motion of the three systems that are considered in this thesis. As explained below, the systems are referred to as *Rotate*, *Translate*, and *Dilate*.

The first system consists of a mass-and-spring oscillator enclosed in a frictionless tube such that there is only one degree of freedom of the mass (Fig. II.A.1). An external agent rotates the tube about a perpendicular axis through the center, which is the equilibrium point of the mass. In the rotating frame of reference of the tube, the equation of the motion of the mass is

$$\ddot{x} + \left[\omega_o^2 - \Omega^2(t)\right]x = 0, \qquad \text{(II.A.1)}$$

where x is the displacement from the center, ω_0 is the frequency of the oscillator in the absence of rotation, and $\Omega(t)$ is the instantaneous angular velocity of the tube.

To derive (II.A.1) we first note that, in the rotating frame of the reference of the tube, four forces act on the mass: the spring force, the centrifugal force, the Coriolis force, and the force of the tube. The force exerted by the springs is

$$F_{spring} = -kx = -m\omega_o^2 x, \qquad \text{(II.A.2)}$$

where k is the spring constant and m is the mass. This is a restoring force, which always opposes the displacement. On the other hand, the centrifugal force is an antirestoring force and is given in general by

$$\vec{F}_{centrifugal} = -m\,\vec{\Omega}x\left(\vec{\Omega}x\vec{r}\right),$$

which, in o ur case, reduces to

$$F_{centrifugal} = -m\Omega^2 x, \qquad \text{(II.A.3)}$$

The Coriolis force is, in general,

$$\vec{F}_{Coriolis} = -2m\,\vec{\Omega}x\vec{v},$$

where $\vec{v}$ is the velocity of the mass in the rotating frame. This force acts transverse to the motion relative to the rotating frame and, in our case, reduces to

$$F_{Coriolis} = -2m\Omega v. \qquad \text{(II.A.4)}$$

The force F_{tube} of the tube is also purely transverse due to the assumed absence of friction. F_{tube} is a reactive force whose instantaneous value is such that the net transverse force accounts for the transverse acceleration $\dot{\Omega}x$ of the mass. Hence, in the transverse direction,

$$F_{tube} + F_{Coriolis} = m\dot{\Omega}x, \qquad \text{(II.A.5)}$$

where $F_{Coriolis}$ is given by (II.A.4). The transverse equation of motion (II.A.5) is irrelevant for our purposes. The equation of motion for the displacement x is found by setting the sum of the spring and centrifugal forces in (II.A.2) and (II.A.3), respectively, equal to the mass m multiplied by the acceleration $\ddot{x}$. The result is the equation of motion (II.A.1).

Regarding the angular velocity $\Omega(t)$ of the tube, we desire this function be identically zero for $t < -T$ and for $t > T$ (where T is a time value of our choice) and to vary

over the time interval [-T,T] such that the tube is rotated one revolution (360°). We also desire that $\Omega(t)$ be as smooth as feasible. A natural choice is thus

$$\Omega(t) = \begin{Bmatrix} 0, & t < -T \\ \Omega_o \operatorname{sech}\left(\frac{\Omega_o t}{2}\right), & -T \le t \le T \\ 0, & t > T \end{Bmatrix}. \qquad \text{(II.A.6)}$$

In this expression, Ω_0 is the peak angular velocity which occurs at t = 0. The near-adiabatic limit corresponds to small values of Ω_0; the abrupt limit corresponds to large values. We choose T to be sufficiently large such that the discontinuity in $\Omega(t)$ at $|t| = T$ is sufficiently small and thus has a negligible effect upon the motion. Furthermore, the normalization of the sech argument in (II.A.6) is chosen to yield (as closely as possible) a single revolution if T is large:

$$\int_{-\infty}^{\infty} \Omega_o \operatorname{sech}\left(\frac{\Omega_o t}{2}\right) dt = 2\pi.$$

Our first model system is governed by (II.A.1) and (II.A.6). This system, as well as the computer program that simulates the motion, is referred to as *Rotate*.

Our second system is identical to the first except that the alteration is a translation along the axis of the tube rather than a rotation about a perpendicular bisecting axis (Fig. II.A.2). The equation of motion is given by

$$\ddot{x} + \omega_o^2 x = -\dot{V}(t), \qquad \text{(II.A.7)}$$

where V(t) is the instantaneous velocity of the tube along its axis.

To derive (II.A.7) we note that, in the frame of reference of the tube,

an inertial force exists in addition to the spring force. By Einstein's Equivalence Principle, the inertial force has magnitude equal to the mass multiplied by the acceleration of the frame (relative to an inertial frame), and is directed opposite to this acceleration. This leads immediately to the equation of motion (II.A.7).

As with the angular velocity in (II.A.6), we choose the linear velocity in the second model to be

$$V(t) = \begin{cases} 0, & t < -T \\ V_o \operatorname{sech}\left(\frac{V_o t}{\pi}\right), & -T \le t \le T \\ 0, & t > T \end{cases}, \qquad \text{(II.A.8)}$$

where the quantity V_O is the peak value of the velocity of the tube. We again choose T sufficiently large to ensure that the discontinuity of V(t) at $|t| = T$ is negligible. We have chosen the argument of sech in (II.A.8) such that the total displacement of the tube is unity:

$$\int_{-\infty}^{\infty} V_o \operatorname{sech}\left(\frac{V_o t}{\pi}\right) dt = 1.$$

Our second model system is governed by (II.A.7) and (II.A.8). We refer to this system and computer program as *Translate*.

Our third equation of motion is a standard one in considerations of adiabatic invariance:

$$\ddot{x} + \omega^2(t)x = 0, \qquad \text{(II.A.9)}$$

where tne frequency ω, which is a function of time, has an initial value ω_0 that is constant for $t < -T$, a final value ω_1 that is constant for $t > T$, and varies at intermediate times. We choose

$$\omega^2(t) = \begin{cases} \omega_o^2, & t < -T \\ \frac{\omega_1^2+\omega_o^2}{2} + \frac{\omega_1^2-\omega_o^2}{2} \tanh(\alpha t), & -T \leq t \leq T \\ \omega_1^2, & t > T \end{cases}. \qquad \text{(II.A.10)}$$

The near-adiabatic limit corresponds to small values of α compared to ω_o and ω_1; the abrupt limit corresponds to large values.

A physical realization of (II.A.9) is shown in Fig. II.A.3. The mass moves parallel to the walls. If the amplitude of the motion remains small, and if the springs always have a nonzero tension, then equal movement of the walls either inward or outward can give rise to the equation of motion (II.A.9). Due to this physical realization, we refer to the system and the computer program as *Dilate*.

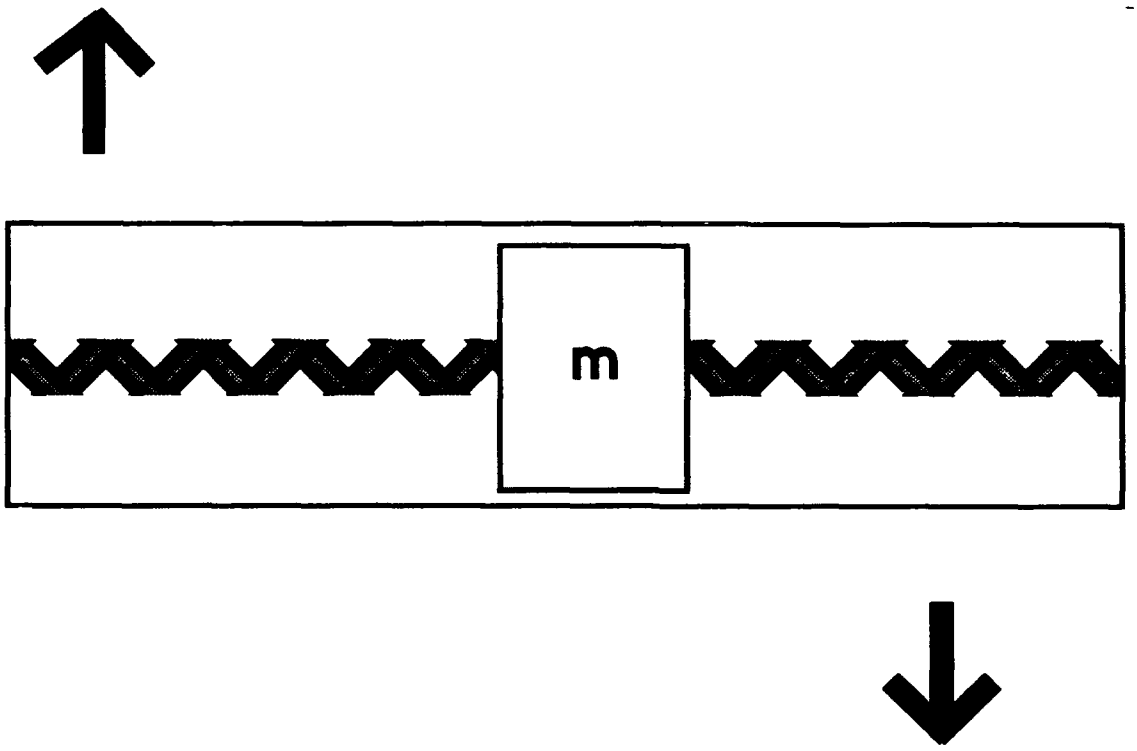


Fig. II.A.1. *Rotate* system. The tube enclosing the mass-and-spring system rotates with angular velocity $\Omega(t)$. The spring constant of the system is k.

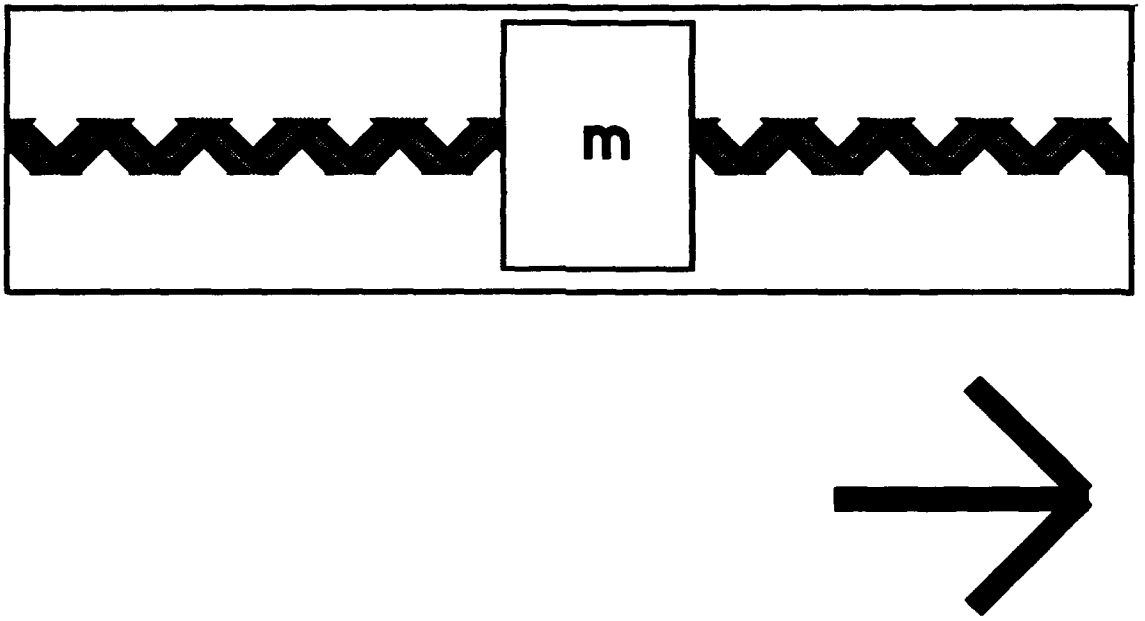


Fig. II.A.2. *Translate* system. The tube enclosing the mass-and-spring system is translated with instantaneous velocity V(t) along its axis. The spring constant of the system is k.

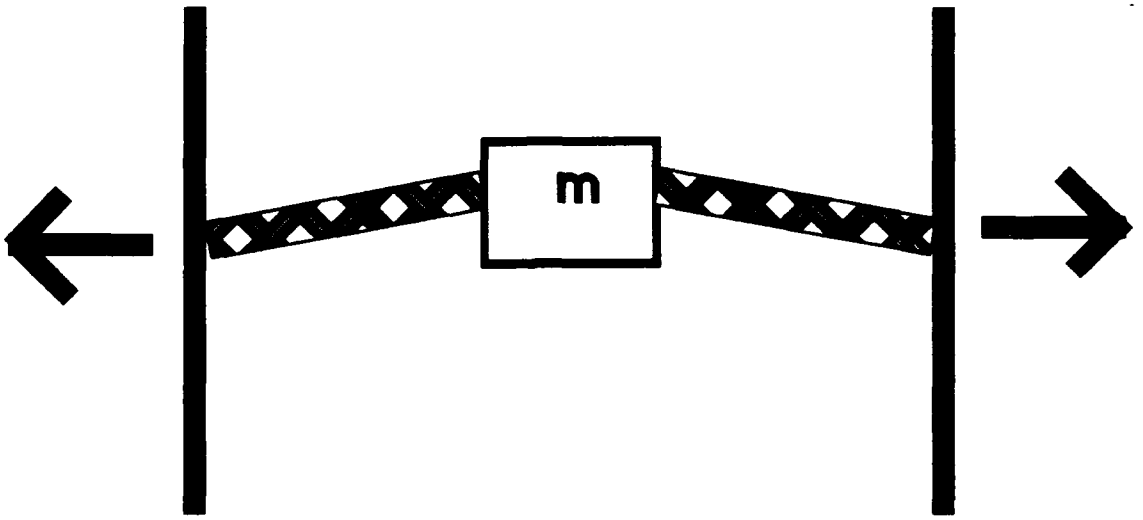


Fig. II.A.3. *Dilate* system. The motion of the mass is transverse to the springs. The walls are both moved either outward or inward.

B. NEAR-ADIABATIC LIMIT

Before we consider the weak breakdown of adiabatic invariance, we examine the adiabatic invariance which occurs in the limit of infinitely slow alterations of a parameter of a system. Under these conditions, it can be shown for any periodic Hamiltonian system that the actions J_i are invariant:

$$J_i = \oint p_i \, dq_i, \tag{II.B.1}$$

where p_i and q_i are the conjugate momentum and position variables of the i^{th} degree of freedom, and where the integral is over one cycle of the motion. If the motion is simple harmonic, (II.B1) reduces to $J = E/\omega$, where E is the energy of the oscillator and ω is the angular frequency. In general, then, for infinitely slow variations of a parameter of a harmonic oscillator,

$$\frac{E}{\omega} = \text{constant}, \tag{II.B.2}$$

This result can be derived explicitly for a variety of oscillators. The most widely known case is a small-amplitude pendulum whose length is slowly altered (Kubo, 1967). We now explicitly derive (II.B.2) for the case of our *Dilate* system (Sec. II.A). To our knowledge, this derivation is not in the literature.

The geometry of the system is shown in Fig. II.B.1. It is not difficult to show that the equation of motion for the position x is

$$m\ddot{x} + 2k_o \frac{L - L_o}{L} x = 0, \tag{II.B.3}$$

which is valid in the limit of small oscillations. We will be concerned only with the case $L > L_0$ (i.e., the springs are always stretched). From (II.B.3) we find that the frequency is given by

$$\omega^2 = \frac{2k_o}{m}\frac{L - L_o}{L}. \qquad \text{(II.B.4)}$$

We now imagine that both walls in Fig. II.B.1 are slowly moved a small distance either outward or inward, so that the new distance between the wall is $L + \delta L$. The frequency will change according to (II.B.4). How does the amplitude (or energy) change? We can calculate this change by employing the conservation of energy. The work done by the external agent that moves the walls is

$$\delta W = 2f\delta L, \qquad \text{(II.B.5)}$$

where the normal force exerted by a spring on a wall is

$$f = < T\cos\theta >, \qquad \text{(II.B.6)}$$

where T is the tension and θ is the angle of the opening from the perpendicular. We have included a time average in (II.B.6) due to the assumption that the displacement of the walls is slow compared to the period of the oscillations. The tension T and angle θ are given by

$$T = k_o\left(\sqrt{L^2 + x^2} - L_o\right),$$

$$\cos\theta = \frac{L}{\sqrt{L^2 + x^2}}.$$

Substituting these into (II.B.6), and approximating to the lowest order in x/L, gives

$$f = k_o(L - L_o) + \frac{k_o L_o}{2L^2} < x^2 >. \qquad \text{(II.B.7)}$$

The total energy of the system is

$$E_{tot} = k_o(L - L_o)^2 + E, \qquad \text{(II.B.8)}$$

where the first term on the right is the rest energy, and the second term is the oscillation energy. The latter is given by

$$E = \frac{1}{2} m\dot{x}^2 + \frac{1}{2} m\omega^2 x^2, \qquad \text{(II.B.9)}$$

The work done by the external agent equals the change in energy of the system:

$$\delta W = \delta E_{tot}, \qquad \text{(II.B.10)}$$

Substituting (II.B.7) into (II.B.5), and then substituting the resultant expression and (II.B.8) into (II.B.10), gives

$$\delta E = \frac{k_o L_o}{L^2} < x^2 > \delta L. \qquad \text{(II.B.11)}$$

From (II.B.9) and the fact that the average kinetic and potential energies are equal, we find

$$E = m\omega^2 < x^2 >,$$

Substituting this into (II.B.11), and rearranging, gives

$$\frac{\delta E}{E} = \frac{k_o L_o}{m\omega^2 L} \frac{\delta L}{L}. \qquad \text{(II.B.12)}$$

By performing the differential of the frequency (II.B.4), we find

$$\frac{\delta L}{L} = \frac{mL}{k_o L_o}\omega\,\delta\omega\,.$$

Finally, substituting this into (II.B.12) and simplifying gives

$$\frac{\delta E}{E} = \frac{\delta\omega}{\omega},$$

which is equivalent to the general result (II.B.2).

For any harmonic oscillator, the action $J = E/\omega$ is constant if a parameter of the system is altered infinitely slowly. The action is thus an adiabatic invariant. What happens if the alteration is slow but occurs over a finite time? Landau and Lifshitz (1976) have shown that, if a parameter is varied slowly and infinitely smoothly from one asymptotically constant value to another, the adiabatic invariant J changes as

$$\Delta J \approx \exp(-\omega/\Omega),$$

where $2\pi/\Omega$ is a characteristic time of the alteration, and ω is the characteristic frequency of the motion. Changes in the adiabatic invariant are thus exponentially suppressed. In particular, as $\Omega \to 0$, $\partial^n J/\partial\Omega^n \to 0$ for all values of n; that is, the function $J(\Omega)$ is infinitely flat at the origin. For alterations that are slow but not infinitely smooth,

$$\Delta J \approx \left(\frac{\Omega}{\omega}\right)^m$$

where m is the order of the lowest discontinuous derivative (Lenard, 1959). It is our desire that the discontinuity in the models, which occurs at $|t| = T$, has a negligible

effect on the motion if the discontinuity is small. We should therefore observe the exponential suppression, although in Ch. III we will see that this is not the case.

In the *Rotate* and *Translate* systems, the initial and final values of the alteration parameters are identical. Hence, the change ΔJ in the adiabatic invariant is proportional to the change ΔE in energy. In our numerical results for *Dilate* and *Translate*, we deal with the change ΔA in amplitude rather than ΔE. It is easily shown, however, that ΔA should also be exponentially suppressed. For the *Dilate* system, the final frequency differs from the initial frequency, and we accordingly deal with $\Delta J = \Delta(E/\omega)$.

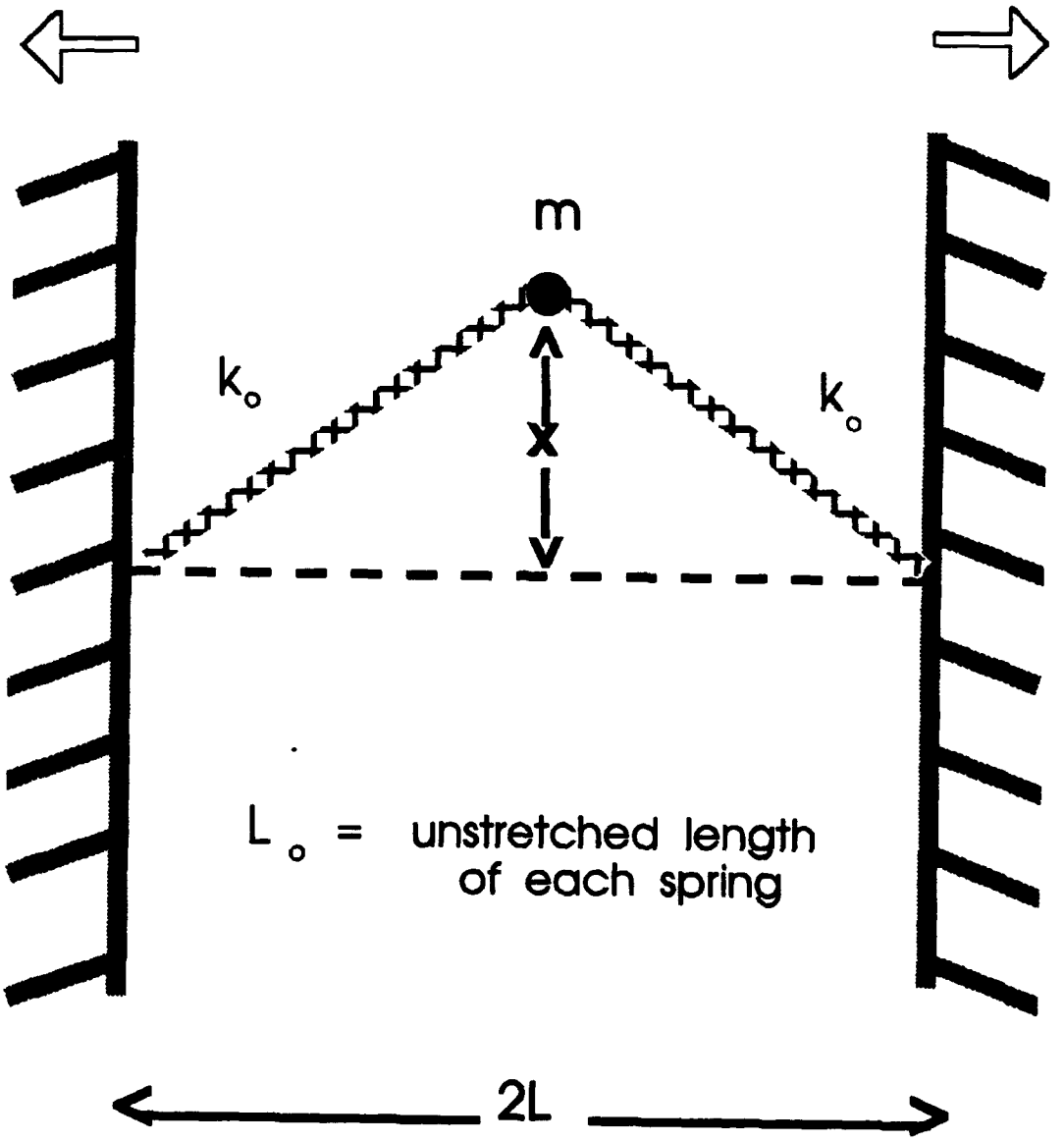


Fig. II.B.1. Geometry of the *Dilate* system. The amplitude of the motion is assumed to be small ($|x| \ll L$).

C. ABRUPT LIMITS

In this section, we analytically solve for the final state of motion of the systems in the limit of an abrupt alterations. In Ch. III, we compare these predictions to numerical results.

We first consider the *Rotate* system. The equation of motion (II.A.1 and II.A.6) can be expressed as

$$\frac{dv}{dt} = \left[\Omega_o^2 \sec h^2\left(\frac{\Omega_o t}{2}\right) - \omega_o^2\right] x, \qquad \text{(II.C.1)}$$

where v is the velocity dx/dt. In the abrupt limit ($\Omega_o >> \omega_o$), the position is approximately

$$x(t) = \begin{cases} A_o \sin(\omega_o t + \phi_o), & t < 0 \\ A_1 \sin(\omega_o t + \phi_1), & t > 0 \end{cases}, \qquad \text{(II.C.2)}$$

where A_o is the initial amplitude and ϕ_o is the initial phase. We wish to determine the final amplitude A_1 and final phase ϕ_1. Continuity of position at t = 0 implies

$$A_o \sin\phi_o = A_1 \sin\phi_1. \qquad \text{(II.C.3)}$$

The velocity at t = 0, however, is discontinuous in the abrupt limit. To determine the jump Δv in velocity, we integrate (II.C.1) over a small time interval about t = 0. This gives, approximately,

$$\Delta v = \Omega_o^2 x(0) \int_{-\infty}^{\infty} \sec h^2\left(\frac{\Omega_o t}{2}\right) dt \;=\; 4\Omega_o x(0), \qquad \text{(II.C.4)}$$

which is asymptotically accurate as $\Omega_o \to \infty$. Combining (II.C.4) and (II.C.2), we find that the velocity jump condition is

$$\omega_o\left(A_1\cos\phi_1 - A_o\cos\phi_o\right) = 4\Omega_o A_o \sin\phi_o. \qquad \text{(II.C.5)}$$

Because $\Omega_o >> \omega_o$, we can neglect the second term in (II.C.5) if $\sin\phi_o \neq 0$. If $\sin\phi_o = 0$, then the right side of (II.C.5) vanishes. Hence, we can approximate (II.C.5) as:

$$\begin{aligned} \omega_o A_1\cos\phi_1 &= 4\Omega_o A_o \sin\phi_o, \quad \text{if } \sin\phi_o \neq 0 \\ A_1\cos\phi_1 &= A_o, \quad \text{if } \sin\phi_o = 0. \end{aligned} \qquad \text{(II.C.6)}$$

The relationships (II.C.3) and (II.C.6) constitute two equations in the two unknowns A_1 and ϕ_1. The solution for $\sin\phi_o \neq 0$ is

$$\begin{aligned} A_1 &= \frac{4\Omega_o}{\omega_o}, \\ \phi_1 &= \tan^{-1}\left(\frac{\omega_o}{4\Omega_o}\right). \end{aligned} \qquad \text{(II.C.7)}$$

The solution for $\sin\phi_o = 0$ is

$$\begin{aligned} A_1 &= A_o, \\ \sin\phi_1 &= 0. \end{aligned} \qquad \text{(II.C.8)}$$

The results (II.C.7) and (II.C.8) are valid in the abrupt limit ($\Omega_o >> \omega_o$). They are compared to the numerical simulations in Sec. III.B.

In the case of the *Translate* system, the position in the abrupt limit is also given by (II.C.2). To determine the final amplitude A_1 and final phase ϕ_1, it is convenient to consider the motion in the laboratory frame of reference. We suppose that the tube abruptly suffers a longitudinal displacement D at t = 0. In the laboratory frame, the force

on the mass is due solely to the springs, and this force remains finite over the vanishingly small time interval of the alteration. The velocity must therefore be continuous. The position relative to the tube, however, is discontinuous by the amount -D, because the mass does not move during the alteration. The position and velocity relationships at t = 0 thus yield, respectively,

$$A_1 \sin\varphi_1 = A_o \sin\varphi_o - D,$$
$$A_1 \cos\varphi_1 = A_o \cos\varphi_o.$$

By adding the squares of each of these relationships to each other, we determine the final amplitude A_1. By dividing the relationships, we determine the final phase ϕ_1. The results are

$$A_1 = \left[A_o^2 - 2A_o D \sin\varphi_o + D^2\right]^{\frac{1}{2}},$$
$$\varphi_1 = \tan^{-1}\left(\frac{A_o \sin\varphi_o - D}{A_o \cos\varphi_o}\right). \qquad \text{(II.C.9)}$$

In Sec. III.C, we will compare these theoretical results to the numerical simulations.

In the case of an abrupt alteration in *Dilate* (Fig. II.C.1), the position of the mass is given by

$$x(t) = \begin{Bmatrix} A_o \sin(\omega_o t + \phi_o), & t < 0 \\ A_1 \sin(\omega_1 t + \phi_1), & t > 0 \end{Bmatrix}, \qquad \text{(II.C.10)}$$

where the final frequency is now different than the initial frequency, in contrast to (II.C.2). The unknowns are the final amplitude A_1 and final phase ϕ_1. To solve for these, we first note that the position must be continuous, which implies that, at t = 0,

$$A_o \sin\phi_o = A_1 \sin\phi_1. \qquad \text{(II.C.11)}$$

Second, the force f on the mass is proportional to $-\omega^2(t)x$, and $\omega(t)$ is always finite by (II.A.10). Hence, the impulse ($\int f\,dt$) over a vanishingly small interval about t = 0 must vanish. This implies that the momentum and, hence, velocity must be continuous. From (II.C.10), the continuity of velocity at t = 0 implies

$$\omega_o A_o \cos\phi_o = \omega_1 A_1 \cos\phi_1. \qquad \text{(II.C.12)}$$

The relationships (II.C.11) and (II.C.12) constitute two equations for the two unknowns A_1 and ϕ_1. Solving for these quantities gives

$$A_1 = A_o\left[\frac{\omega_o^2}{\omega_1^2} + \left(1 - \frac{\omega_o^2}{\omega_1^2}\right)\sin^2\phi_o\right]^{\frac{1}{2}} \qquad \text{(II.C.13)}$$

$$\phi_1 = \tan^{-1}\left(\frac{\omega_1}{\omega_o}\tan\phi_o\right). \qquad \text{(II.C.14)}$$

The change in adiabatic invariant is

$$\Delta\left(\frac{E}{\omega}\right) = \frac{E_1}{\omega_1} - \frac{E_o}{\omega_o} = \frac{1}{2}\omega_1 A_1^2 - \frac{1}{2}\omega_o A_o^2.$$

Substituting (II.C.13), and simplifying, gives

$$\Delta\left(\frac{E}{\omega}\right) = \frac{1}{2}\omega_o A_o^2\left[\frac{\omega_o}{\omega_1} + \left(\frac{\omega_1}{\omega_o} - \frac{\omega_o}{\omega_1}\right)\sin^2\phi_o - 1\right]. \qquad \text{(II.C.15)}$$

This expression gives the predicted change in adiabatic invariant in the abrupt limit of the *Dilate* system. In Sec. III.D, we compare the predicted change in adiabatic invariant to numerical data.

In summary, the effects of abrupt alterations in the three models are characterized as follows. In *Rotate*, the position is continuous while the velocity is discontinuous. The reverse occurs in *Translate*, where the position is discontinuous and the velocity is continuous. Finally, in *Dilate* both the position and velocity are continuous.

III. NUMERICAL SIMULATIONS

A. IMPLEMENTATION

The numerical simulation programs were implemented using the Euler-Cromer method (Gould and Tobochnik, 1988) of approximating ordinary differential equations, as we now explain. It should be noted that Runge-Kutta methods are unstable for periodic autonomous motion in a Hamiltonian system.

For motion with one degree of freedom, we assume that the acceleration a is a known function of the position x, velocity v, and time t. Hence,

$$\frac{dv}{dt} = a(x, v, t),$$
$$\frac{dx}{dt} = v.$$

The velocity v_{n+1} and position x_{n+1} at time t_{n+1}, where $t_{n+1} = t_n + \Delta t$, can then be approximated as

$$\begin{aligned} v_{n+1} &= v_n + a_n \Delta t, \\ x_{n+1} &= x_n + v_{n+1} \Delta t, \end{aligned} \qquad \text{(III.A.1)}$$

where $a_n = a(x_n, v_n, t_n)$. The standard Euler method employs the old velocity v_n in the second equation, and is unstable. The use of the updated velocity v_{n+1} causes the method to be stable.

The final states of the systems depend upon the initial phase. We define phase as $\tan^{-1}(x/v)$, where x is the position and v is the velocity. The initial phase (when the alteration of the system is initiated and is very small) is constructed to be identical to the

phase at the zero of time for the case of no alteration. The zero of time corresponds to the maximum alteration in the case of *Rotate* and *Translate*, and to the maximum rate of change of the alteration in the case of *Dilate*. In the first two cases, the system parameters are returned to their initial values. Hence, relative to the initial state of motion, the final state can be characterized not only by the change in amplitude, but also by the change in final phase. This quantity equals the final phase (when the alteration of the system is concluded and is very small) minus the value of the phase at this time in the case of no alteration. By construction, the latter is identical to the initial phase.

Ideally, we would like for the alteration of a system to be identically zero outside a time interval (-T,T), and nonzero and infinitely smooth inside the interval. Unfortunately, such an analytic function does not exist. This is demonstrated by a Taylor expansion at $|t| = T$ which shows that the function must be constant everywhere. Our approach, as stated in Sec. II.A, is to choose an infinitely smooth function which is exponentially localized in time, and to choose the value of T sufficiently large such that the difference between the function and a constant is sufficiently small. The constant in *Rotate* and *Translate* is zero while the difference in *Dilate* is nonzero. This small difference is labeled ε, which is specified by the program user. The total time T is then calculated with asymptotic expressions of the alteration functions for large times (because T is large). For *Rotate*, we find from (II.A.6)

$$T = \frac{2}{\Omega_o} \log_e \frac{2\Omega_o}{\varepsilon}. \qquad \text{(III.A.2)}$$

Next, this value is increased by the smallest amount such that the new value corresponds to an integral number of cycles when no alteration occurs. This has the advantage of giving a physical meaning to the initial phase (at t = -T): The initial phase is identical to

the phase at t = 0 and t = T when no alteration of the system occurs. This is important since we wish to examine how the state of motion changes for different values of Ω_o while the initial phase remains fixed. A method such as ours is the only meaningful way of defining initial phase for different values of Ω_o. If $[f]_{max}$ represents the maximum integer contained in the quantity f, the time window (-T,T) for the *Rotate* system model is then given by

$$T = 2\pi\left\{\left[\frac{1}{2\pi}\left(\frac{2}{\Omega_o}\log_e\frac{2\Omega_o}{\varepsilon}\right)\right]_{max} + 1\right\}. \qquad \text{(III.A.3)}$$

In this expression, we have assumed that the period of the motion of the unaltered system equals 2π (i.e., $\omega_o = 1$). The number of cycles corresponding to the preliminary value (III.A.1) is then the preliminary value divided by 2π.

For *Translate*, we differentiate (II.A.8) (since V'(t) occurs in the equation of motion). We then employ asymptotic expansions of the hyperbolic function, and solve for T as a function of ε as in *Rotate* above. This procedure yields the preliminary result

$$T = \frac{1}{V_o}\log_e\frac{2V_o^2}{\pi\varepsilon}. \qquad \text{(III.A.4)}$$

This expression is then adjusted to correspond to a integral number of cycles of the unaltered system, yielding

$$T = 2\pi\left\{\left[\frac{1}{2\pi}\left(\frac{1}{V_o}\log_e\frac{2V_o^2}{\pi\varepsilon}\right)\right]_{max} + 1\right\}. \qquad \text{(III.A.5)}$$

For *Dilate*, we follow the same procedure as in *Rotate* and *Translate*. From (II.A.10), the preliminary expression is

$$T = \frac{1}{2\alpha} \log_e \frac{\left|\omega_1^2 - \omega_o^2\right|}{\varepsilon}, \qquad \text{(III.A.6)}$$

where for clarity we have restored the use of ω_o (rather than setting it equal to unity). The adjusted value for T is then

$$T = \frac{2\pi}{\omega_o} \left\{ \left[\frac{\omega_o}{2\pi} \left(\frac{1}{2\alpha} \log_e \frac{\left|\omega_1^2 - \omega_o^2\right|}{\varepsilon} \right) \right]_{max} + 1 \right\}. \qquad \text{(III.A.7)}$$

The time step Δt in (III.A.1) must be small compared to both the typical period of the motion and the characteristic time over which the alteration occurs. This is important because we desire to obtain data over the complete range of alteration times, from very slow to very fast compared to the period of the motion. The program user effectively enters the value of Δt by specifying the "number of steps per cycle." By "cycle" is meant the period of the unaltered motion or the time of alteration, whichever is smaller. The programs compute this.

B. RESULTS OF ROTATION MODEL

The rotational system simulations consist of three regimes: (a) the abrupt regime, in which the peak angular velocity Ω_0 is substantially greater than unity (i. e., Ω_0 is substantially greater than the natural frequency ω_0 which we set to unity), (b) the intermediate regime, in which Ω_0 is of the order of unity, and (c) the near-adiabatic regime, in which Ω_0 is substantially less than unity. We examine each of these regimes in turn.

Figs. III.B.1 and III.B.2 show the results for the abrupt regime. The oscillator has an initial amplitude of unity, and various values of the initial phase (Sec. III.A). Remarkable behavior characterizes the abrupt regime. For any initial phase between approximately 0° and 60°, the final state of the oscillator is the rest state (the change of amplitude equals -1) for a particular value of the peak rotational angular velocity Ω_0. The final phase suffers an abrupt, although continuous, transition by 180°. To understand this behavior, it is convenient to consider a fixed value of Ω_0 and continuous values of the initial phase. Fig. III.B.3 shows such a case. Fig. III.B.4 shows the corresponding time series (displacement vs. time) for three initial phase values near the transition. Note that the phases are such that the centrifugal force decreases the energy of the oscillator. The centrifugal force in this abrupt case alters the upright potential energy curve such that the curve is momentarily inverted (during a small time interval about $t = 0$). With this picture, we can readily understand the behavior. For one value of the initial phase (near 40.38° in Fig. III.B.4), the work done by the centrifugal force equals the initial energy, and so the oscillator is at rest in the final state. For initial phases slightly less than this, the mass does not pass through the spatial origin when $t \approx 0$ due to the greater effect of the centrifugal

force. For initial phases slightly greater, the mass passes through the origin due to the lesser effect of the centrifugal force. This accounts for the 180° difference in final phase.

The theory for the final amplitude and phase of the *Rotate* system in the abrupt limit is developed in Sec. II.C. According to (II.C.6), for $\sin\phi_0 \neq 0$ the change in amplitude should be asymptotically proportional to the peak rotational angular velocity Ω_0 with a proportionality constant that is independent of the initial phase ϕ_0. Figs. III.B.1a and III.B.2a indeed show an approximately linear relationship in the abrupt regime, but the slope depends substantially upon the initial phase. It may be that the curves approach a common slope for extremely large values of Ω_0, but the theory is too crude for moderately large values of Ω_0. Regarding the final phase, (II.C.7) predicts that this should approach zero in the abrupt limit, so the change in final phase should approach the negative of the initial phase. Figs. III.B.1b and III.B.2b show that this is indeed the case.

Figs. III.B.5 and III.B.6 show the results for the intermediate regime. The change in amplitude appears to approach zero in qualitative accord with the theory (Sec. II.B); that is, the curve becomes very flat (or exponentially suppressed). Furthermore, for smaller values of the peak rotational angular velocity Ω_0, the results appear to become independent of the value of the initial phase. This is in accord with the theory.

However, Figs. III.B.7 and III.B.8 show the remarkable result that the change in amplitude is <u>not</u> exponentially suppressed in the near-adiabatic regime. Rather, the change in amplitude is roughly linear in the alteration parameter Ω_0. Moreover, the dependence of the change in amplitude upon initial phase persists down to the adiabatic limit, although the change in final phase does become independent of the initial phase.

Suspicion of the numerical results for very slow changes of the system is natural. Indeed, <u>any</u> numerical method will eventually become inaccurate for sufficiently slow changes as a result of the large number of simulated cycles and the smallness of the change

in motion of the system. For this reason, the near-adiabatic results should be subjected to further scrutiny, including verification of independence of the "glitch" amount ε and the time step Δt.

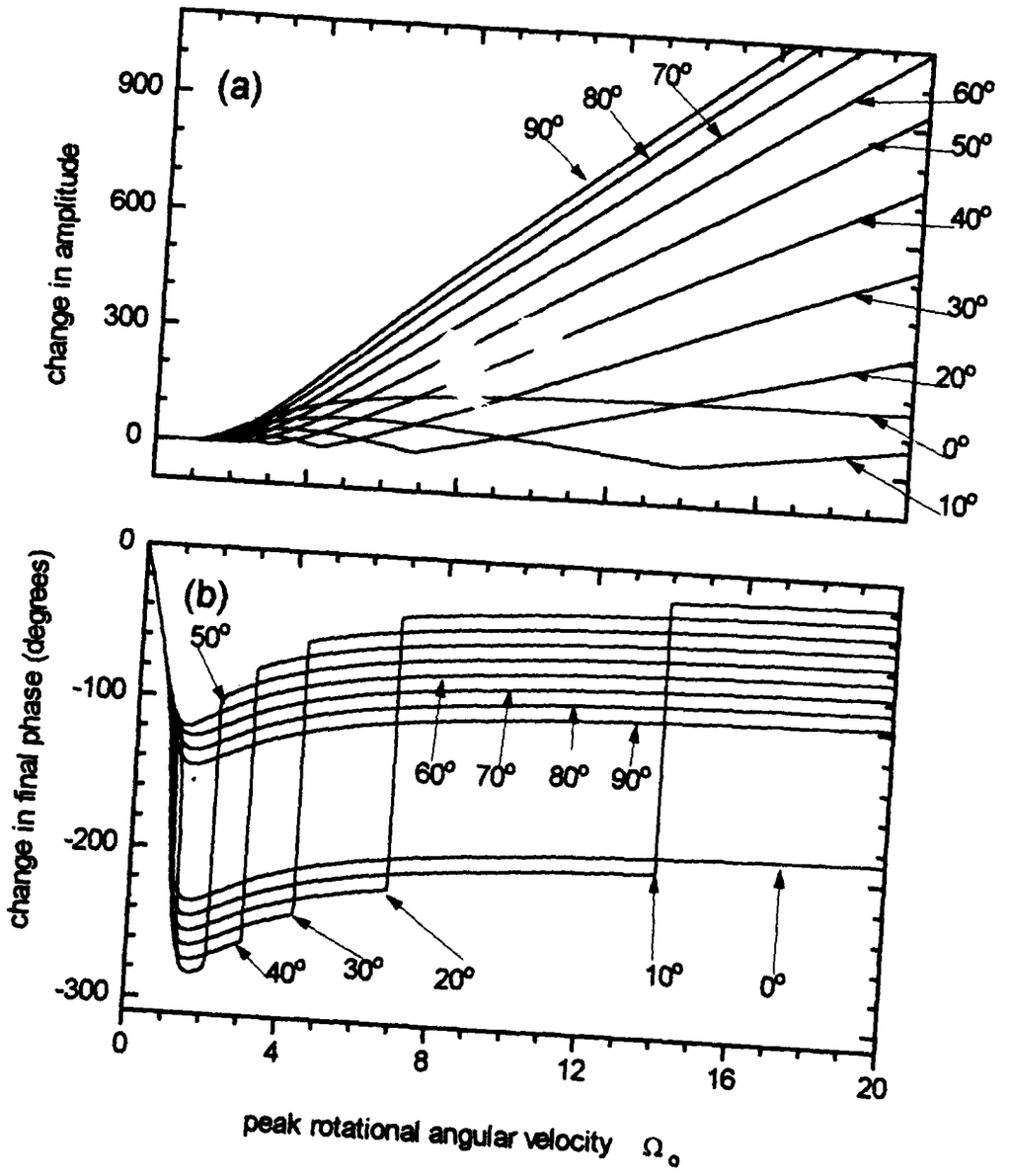


Fig. III.B.1 Numerical simulation results for the *Rotate* model for peak rotational angular velocities Ω_o in the abrupt regime. The various values of the initial phase are shown with each curve.. The numerical parameters are epsilon $\varepsilon = 1 \times 10^{-6}$ and number of steps per cycle = 10,000.

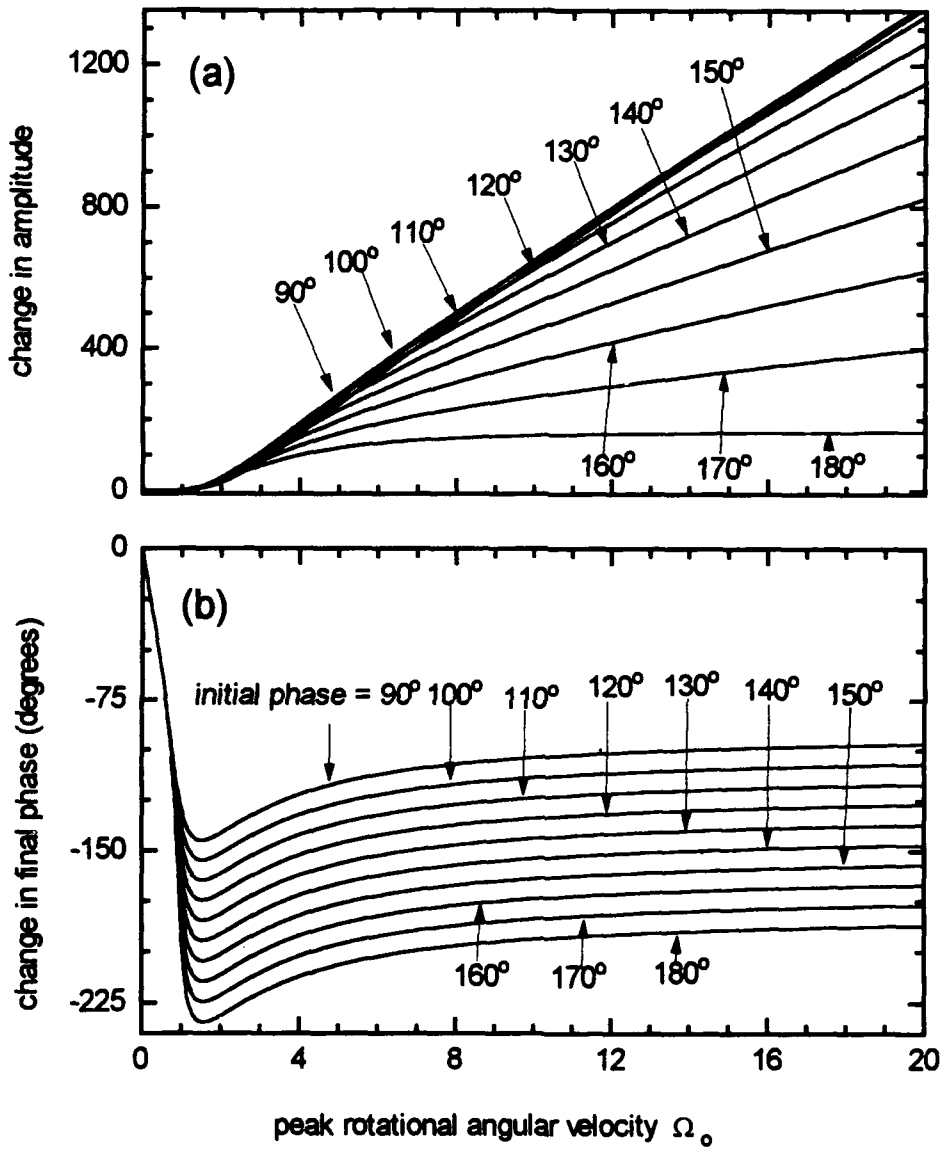


Fig. III.B.2 Numerical simulation results for the *Rotate* model for peak rotational angular velocities Ω_0 in the abrupt regime. The various values of the initial phase are shown each curve.. The numerical parameters are the same as in Fig. III.B.1.

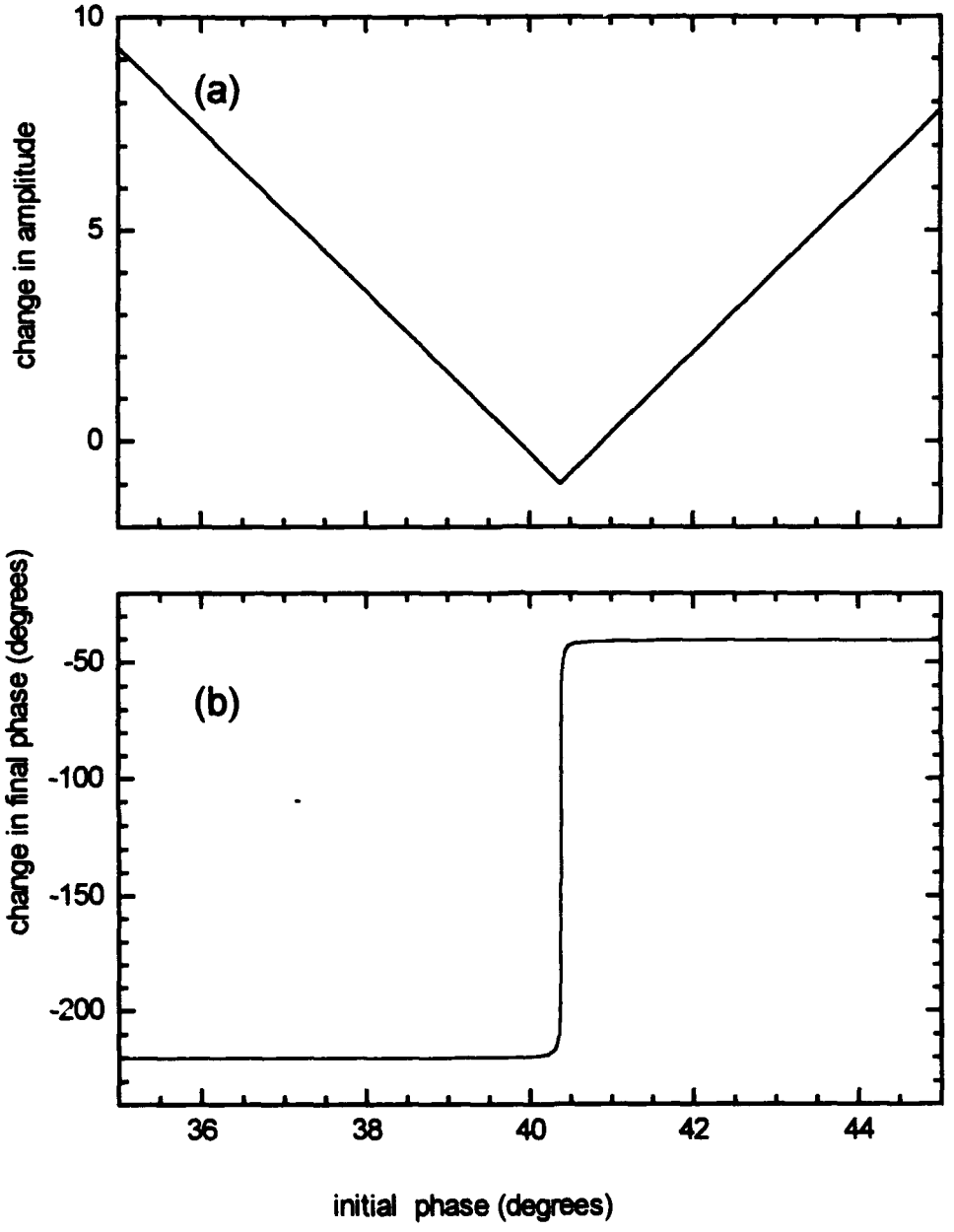


Fig. III.B.3. Numerical simulation of *Rotate* model limit for a fixed peak angular velocity $\Omega_0 = 3$. The values of ε and the number of steps per cycle are the same as in Fig III.B.1.

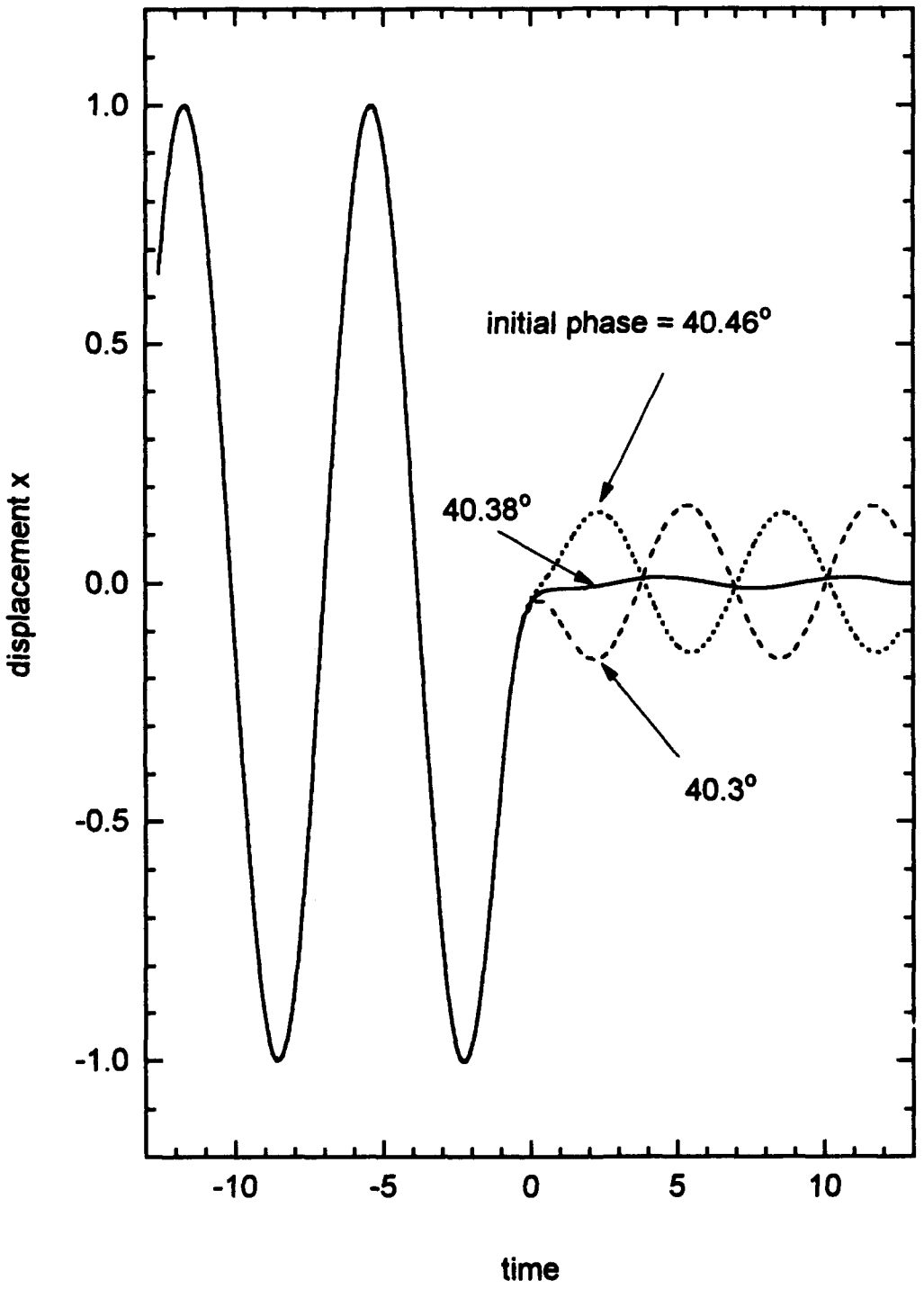


Fig. III.B.4. Displacement in the *Rotate* model for three different initial phase values near the transition. The peak angular velocity is $\Omega_0 = 3$. The values of ε and the number of steps per cycle are the same as in Fig. III.B.1.

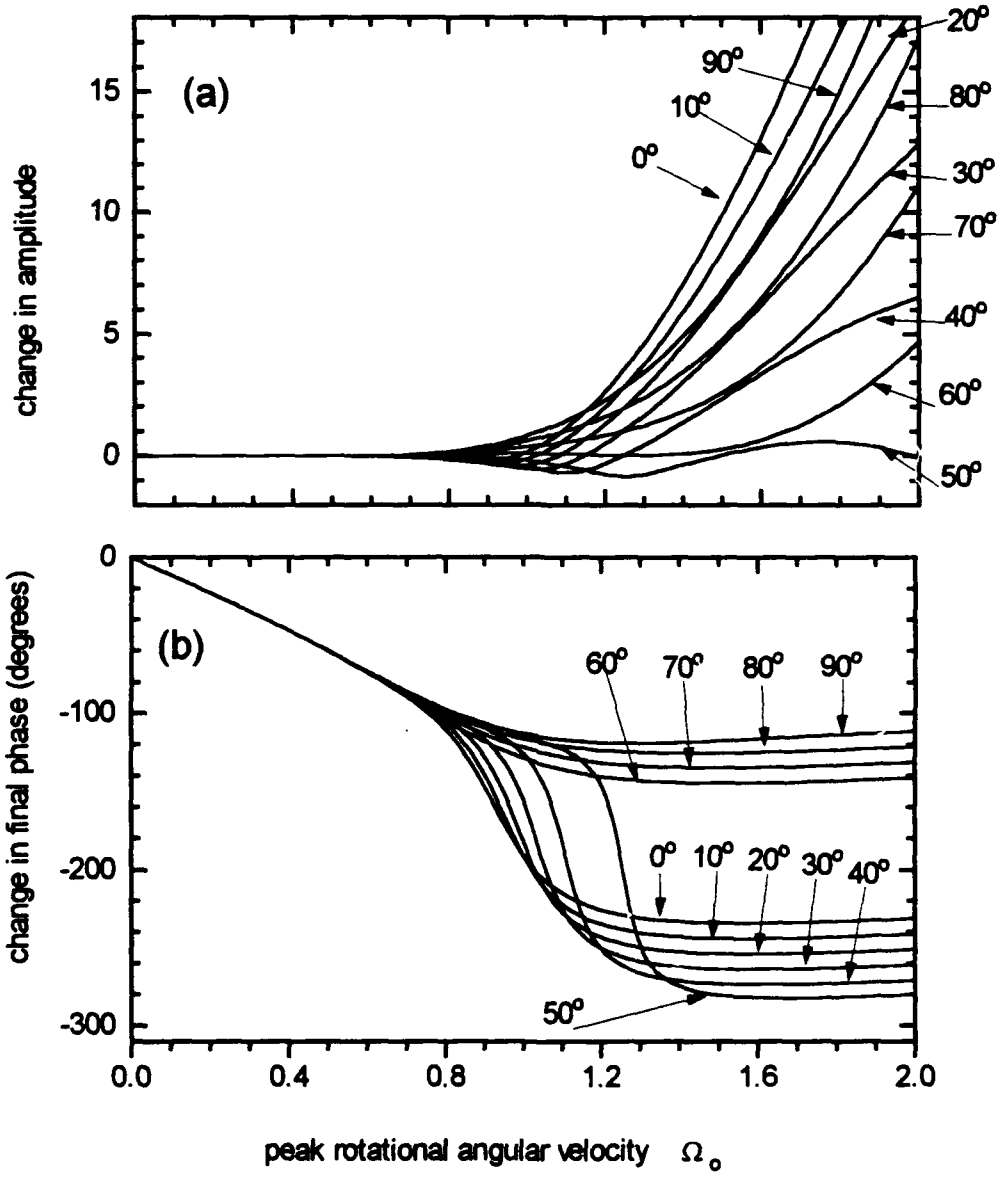


Fig. III.B.5 Numerical simulation results for the *Rotate* model for peak rotational angular velocities Ω_o in the intermediate regime. The various values of the initial phase are shown with each curve.. The numerical parameters are the same as in Fig. III.B.1.

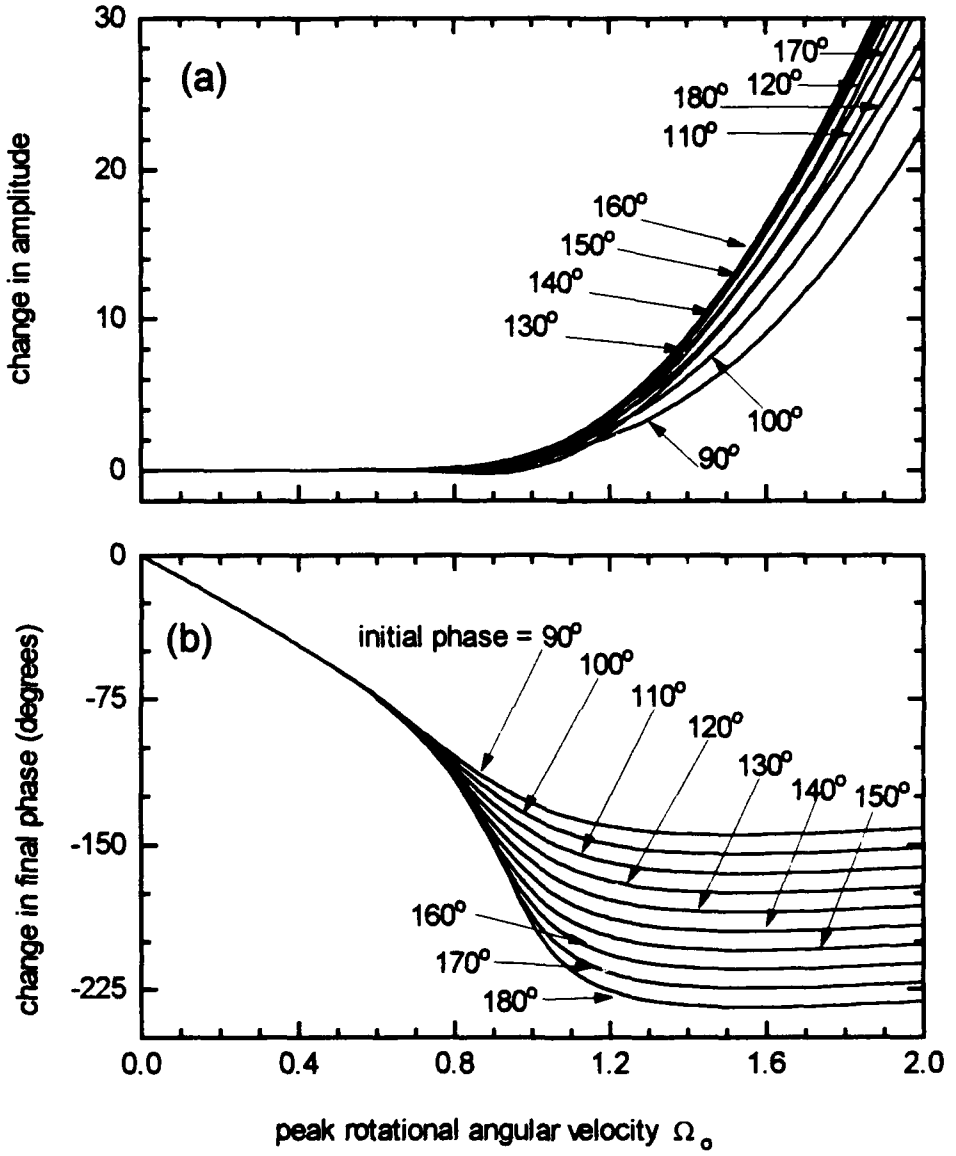


Fig. III.B.6 Numerical simulation results for the *Rotate* model for peak rotational angular velocities Ω_o in the intermediate regime. The various values of the initial phase are shown with each curve.. The numerical parameters are the same as in Fig. III.B.1.

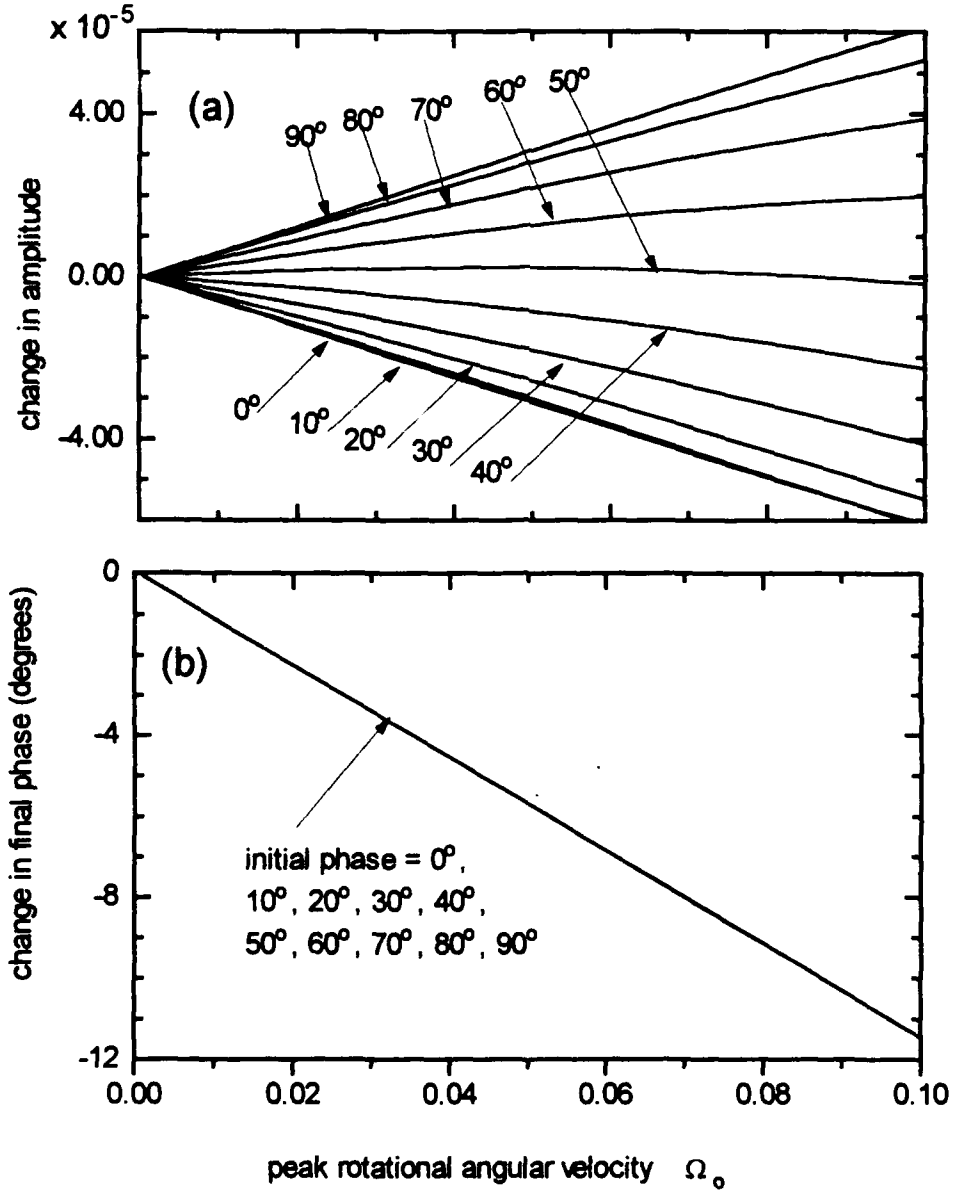


Fig III.B.7. Numerical simulation results for the *Rotate* model for peak rotational angular velocities Ω_0 in the near-adiabatic regime. The various values of the initial phase are shown with each curve.. The numerical parameters are the same as in Fig. III.B.1.

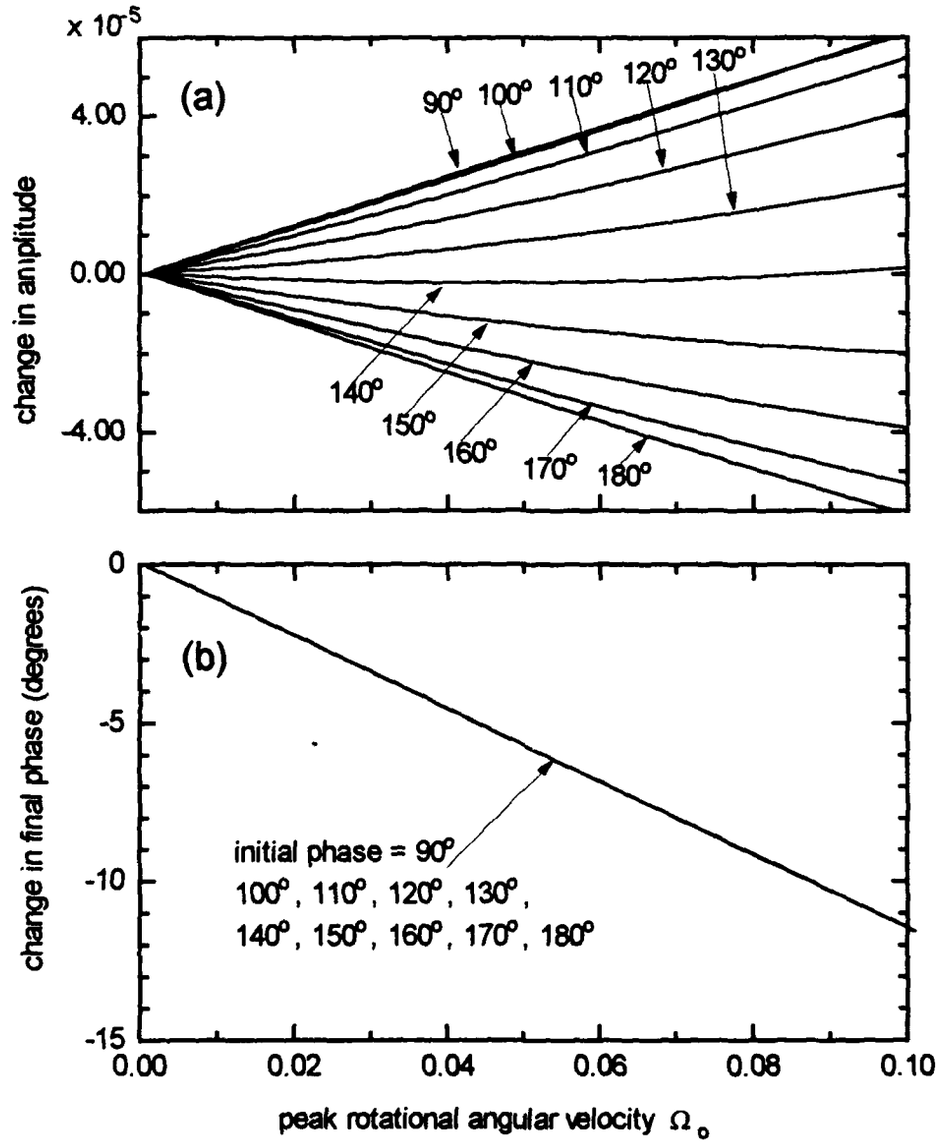


Fig. III.B.8. Numerical simulation results for the *Rotate* model for peak rotational angular velocities Ω_0 in the near-adiabatic regime. The various values of the initial phase are shown with each curve. The numerical parameters are the same as in Fig. III.B.1.

C. RESULTS OF TRANSLATION MODEL

As in the case of *Rotate*, the *Translate* model yielded three regimes with respect of the rate of alteration: abrupt, intermediate, and near-adiabatic. We consider each of these in turn.

Figs. III.C.1 and III.C.2 show changes in amplitude and final phase for values of the peak translational velocity V_0 that include the abrupt limit. A comparison of the numerical results for the changes in amplitude yield excellent agreement with the theoretical predictions (II.C.9) for $A_0 = 1$ and $D = 1$. For an initial phase of 90°, note that the final amplitude is zero. This is easily explained if we consider the motion in the laboratory frame of reference. At $t = 0$, when the mass is at its unit amplitude turning point in the positive direction, the tube is abruptly translated one unit in the positive direction. The mass is then at the equilibrium point with zero velocity, and will thus remain at rest.

An interesting feature of the changes in amplitude in Figs. III.C.1a and III.C.2a is how rapidly the abrupt-limit values are reached as the peak translational velocity V_0 is increased. Even though the characteristic speed of the mass is unity, V_0 need only be roughly equal to 5 to 10 in order for the alteration to be accurately considered as abrupt.

Regarding the abrupt limit of the change in final phase, Fig. III.C.1b shows 360° jumps at various values of the peak translational velocity. These appear to be numerical artifacts of the phase computation, although the problem has not yet been identified. Nearly all of the numerical values in the abrupt limit agree with the theoretical predictions (II.C.9) to within a difference of 360°. In Fig. III.C.2b, nearly all of the numerical values agree with the theoretical values if 180° is added to the latter. Such a deviation is common when the inverse tangent function is employed since it is multivalued by that

increment. Regardless of the ambiguity of 180°, however, the 0° and 90° initial phase data do not agree with the theoretical predictions. Further work is required here.

Figs. III.C.3 and III.C.4 show the intermediate regime. The suppression of the changes in amplitude is dramatic. Furthermore, the changes in final phase are also dramatically suppressed, in contrast to the *Rotate* system.

Figs. III.C.5 and III.C.6 show the near-adiabatic regime. The variations in the change in amplitude indicate significant numerical error. Moreover, the fact that the change in final phase appears to diverge in the adiabatic limit, rather than approach zero as it should, strongly suggests numerical error. The improvement of the computer program here is a subject of future work (Ch. IV).

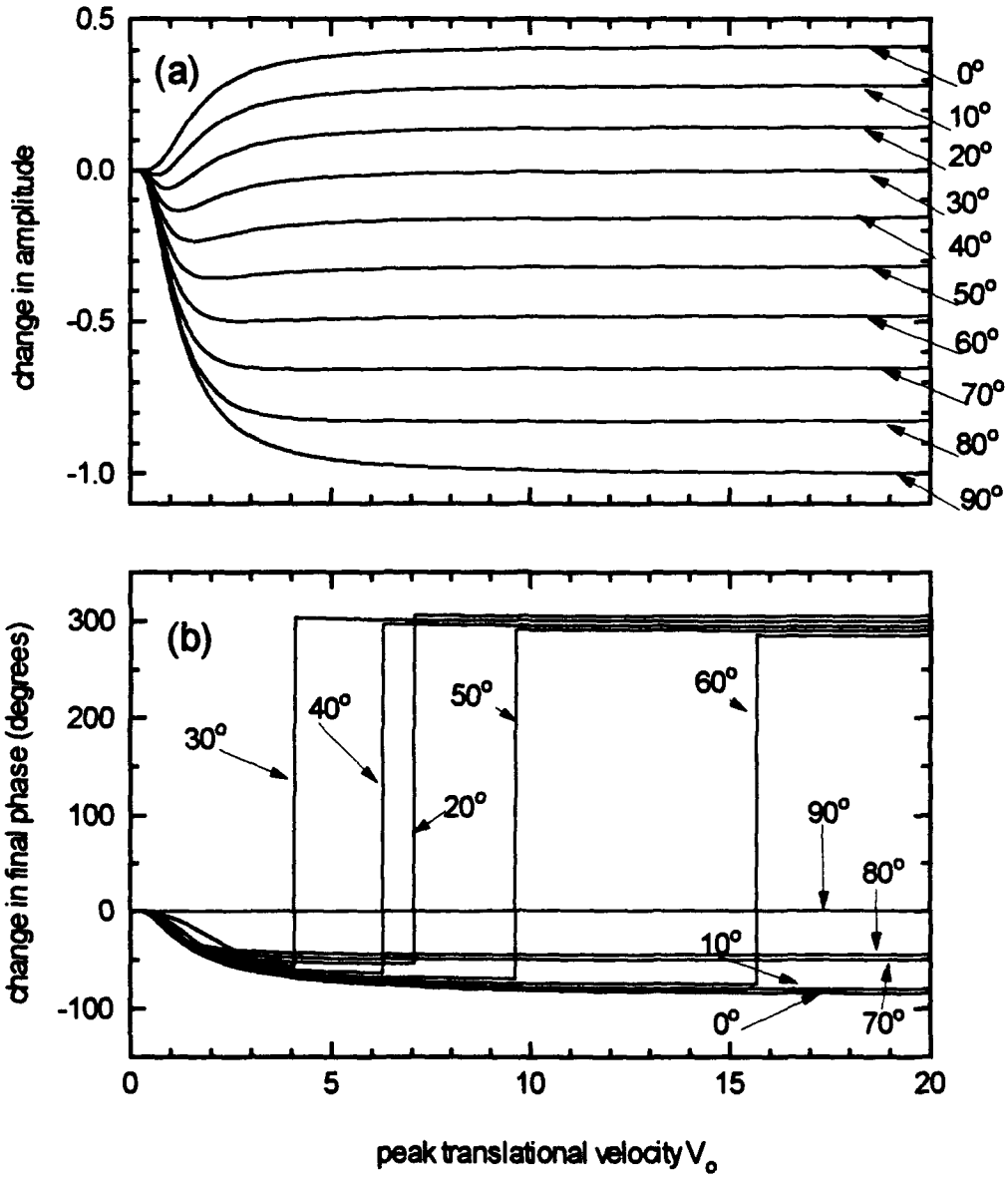


Fig. III.C.1 Numerical simulation results for the *Translate* model in the abrupt regime. Values of the initial phase are displayed near each curve. The numerical parameters are $\varepsilon = 10^{-6}$ and number of steps per cycle = 5000.

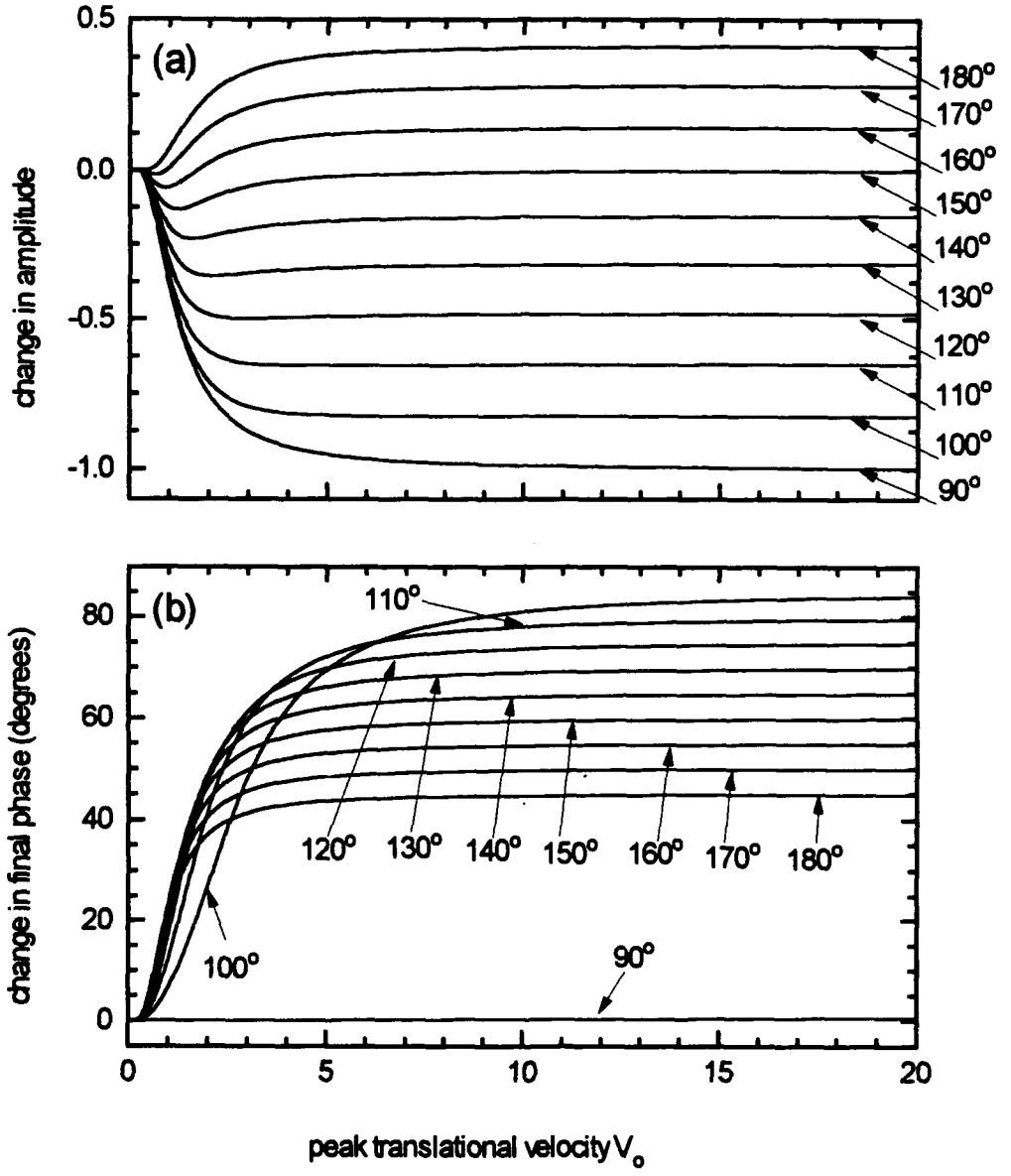


Fig. III.C.2. Numerical simulation results for the *Translate* model in the abrupt regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.C.1.

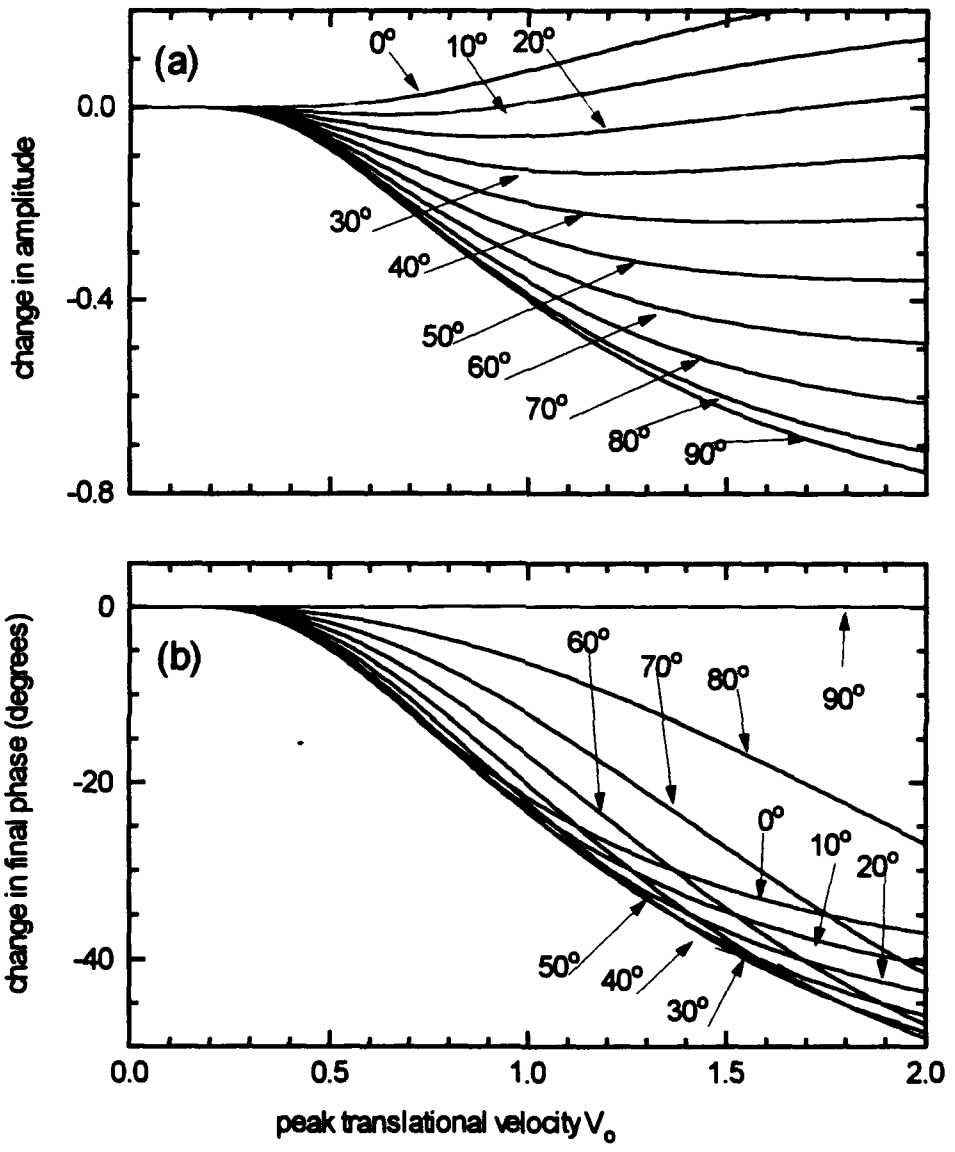


Fig. III.C.3. Numerical simulation results for the *Translate* model in the intermediate regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.C.1.

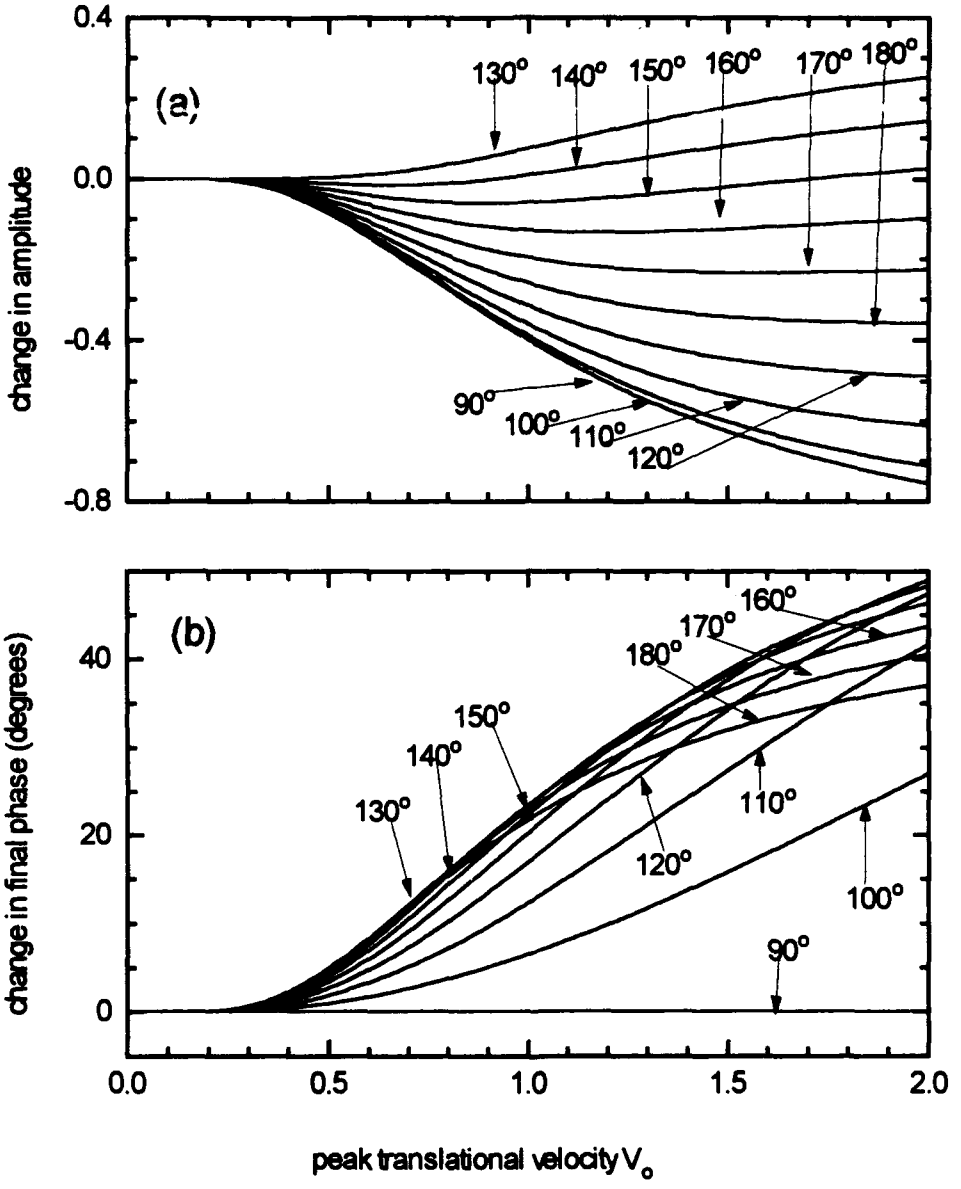


Fig. III.C.4. Numerical simulation results for the *Translate* model in the intermediate regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.C.1.

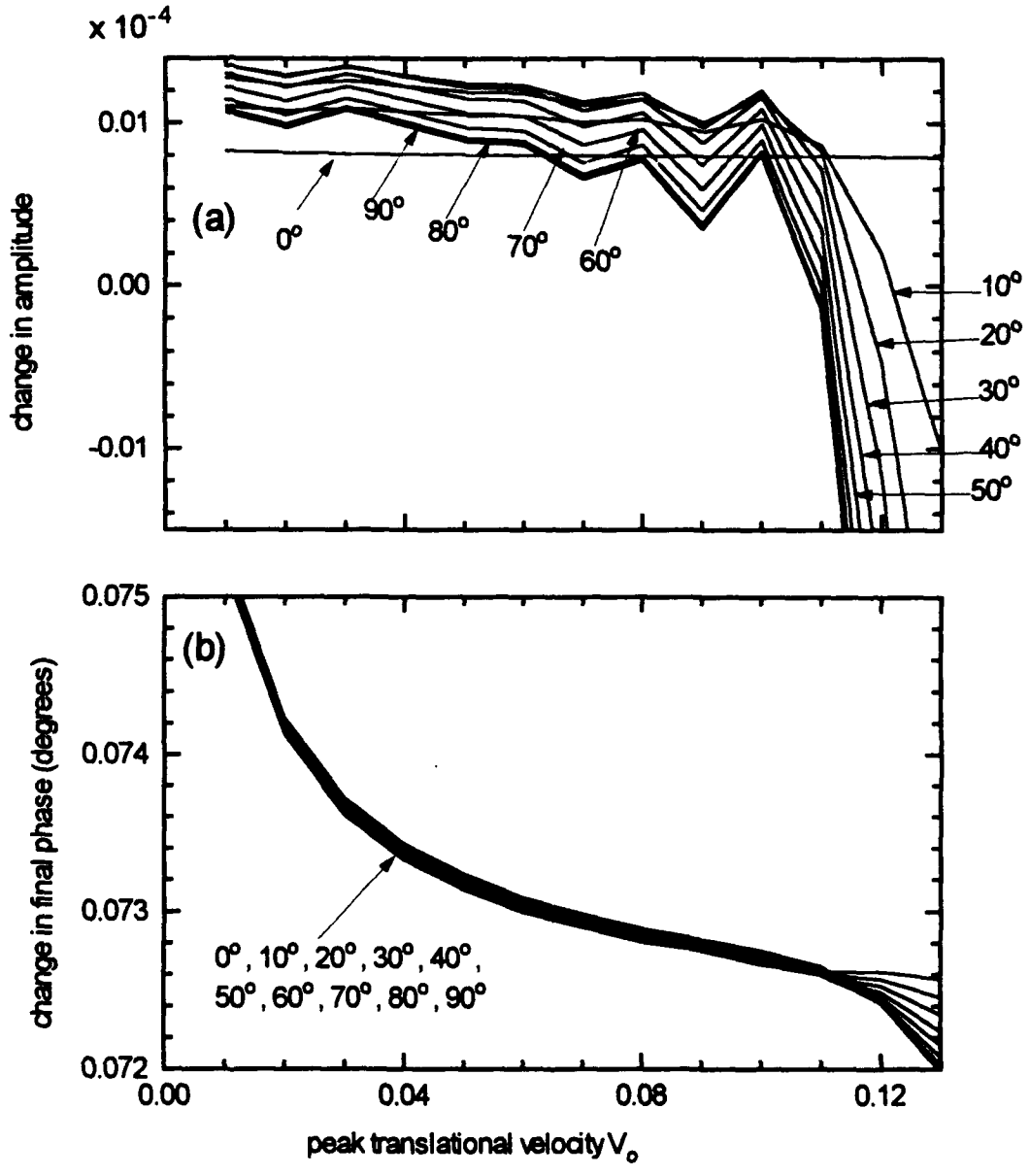


Fig. III.C.5 Numerical simulation results for the *Translate* model in the near-adiabatic regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.C.1.

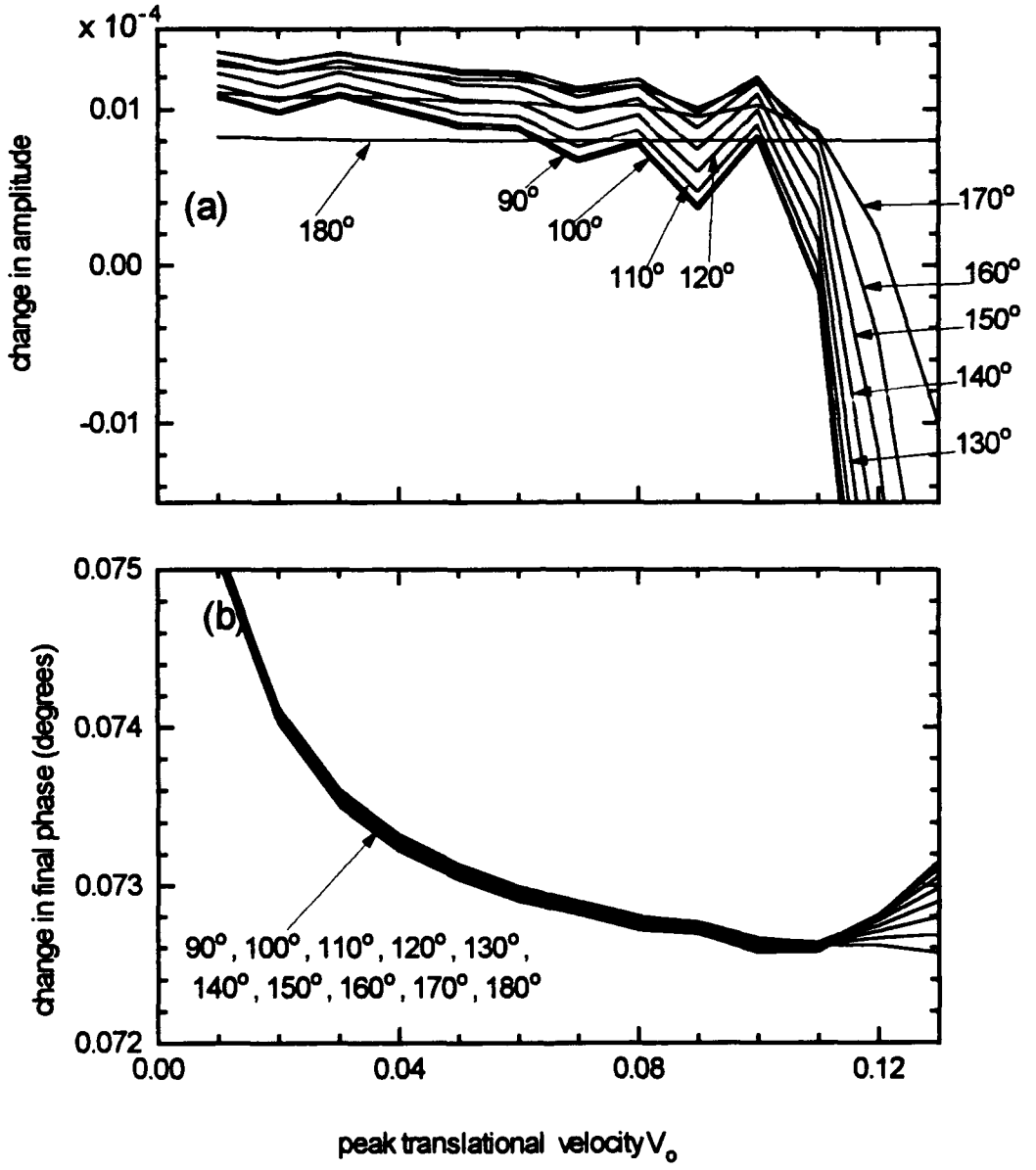


Fig. III.C.6 Numerical simulation results for the *Translate* model in the near-adiabatic regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.C.1.

D. RESULTS OF DILATION MODEL

As in the cases of *Rotate* and *Translate*, the *Dilate* model yielded three regimes with regard to the rate of alteration: abrupt, intermediate, and near-adiabatic. Because the initial and final frequencies are not equal in the case of *Dilate*, we consider the change in adiabatic invariant, rather than the change in amplitude as in *Rotate* and *Translate*. Furthermore, the inequality of the initial and final frequencies implies that a comparison of the final phase to the unaltered case is not meaningful.

Figs. III.D.1 and III.D.2 show the change $\Delta(E/\omega)$ in adiabatic invariant as a function of the rate-of-alteration parameter α on a scale that includes abrupt alterations, ($\alpha >> 1$). That the adiabatic invariant is E/ω is shown very dramatically: All of the curves rapidly approach the origin as α is decreased through the "transition" region from $\alpha \approx 5$ to $\alpha \approx 1$. For increasing values of α, the change in the adiabatic invariant rapidly approaches a constant that depends upon the initial phase. Comparison of the asymptotic values of $\Delta(E/\omega)$ for $\alpha >> 1$ in the figures yields excellent agreement with those predicted by the theoretical expression (II.C.15) for $A_0 = 1$, $\omega_0 = 1$, and $\omega_1 = 2$.

Figs. III.D.3 and III.D.4 show the results in the intermediate regime. The suppression of the change in adiabatic invariant is very dramatic. It is interesting that, whereas the abrupt limit is insensitive to the initial phase transformation $\varphi_0 \rightarrow 180° \rightarrow \phi_0$, as expected on physical grounds, the intermediate/near-adiabatic regime is approximately insensitive to the simultaneous transformation $\varphi_0 \rightarrow \phi_0 + 90°$ and $\Delta(E/\omega) \rightarrow -\Delta(E/\omega)$. The significance of this is not yet known.

The transformation becomes exact in the near-adiabatic regime, as shown in Figs. III.D.5 and III.D.6. More importantly, however, the change in the adiabatic invariant does not approach zero in the adiabatic limit, but approaches a constant that depends upon the

initial phase. This in qualitative accord with Lenard's result (Sec. II.B) for nonsmooth alteration functions, due to the ε "glitch" in our function. By performing computer runs in which the alteration was zeroed out but all other computations were the same, we have verified that the behavior is not a result of the accumulation of error over the many cycles. As a next step, we should decrease ε to observe if the values of $\Delta(E/\omega)$ decrease.

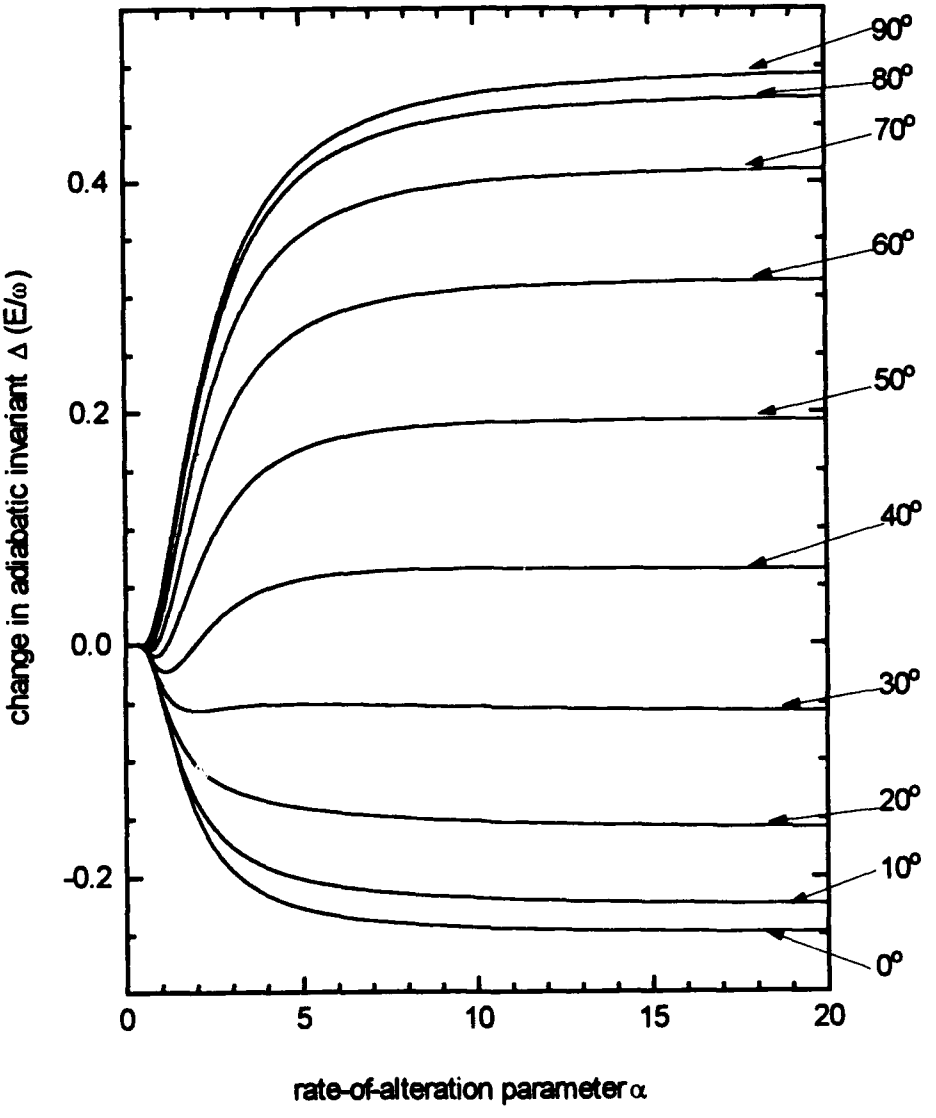


Fig. III.D.1 Numerical simulation results for the *Dilate* model in the abrupt regime. Values of the initial phase are displayed near each curve. The numerical parameters are $\varepsilon = 10^{-6}$ and number of steps per cycle = 10,000.

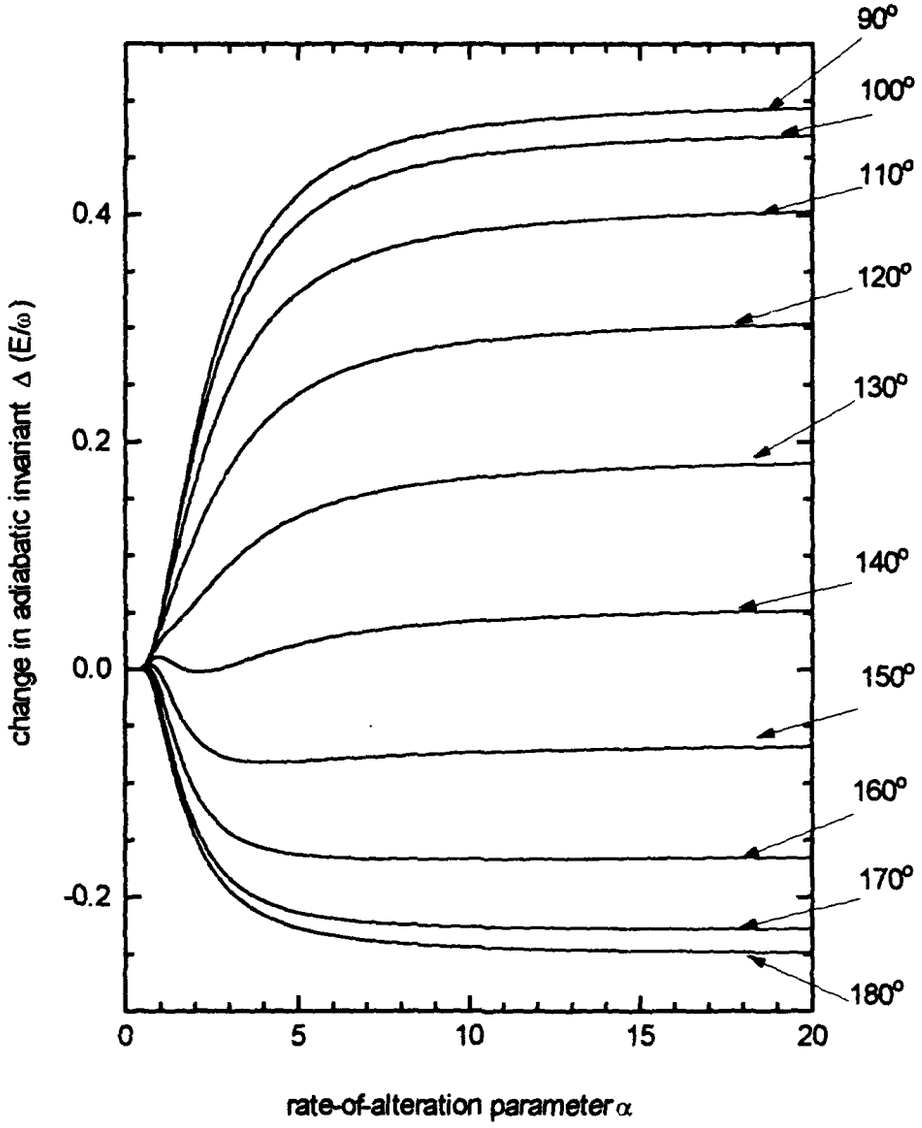


Fig. III.D.2 Numerical simulation results for the *Dilate* model in the abrupt regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.D.1.

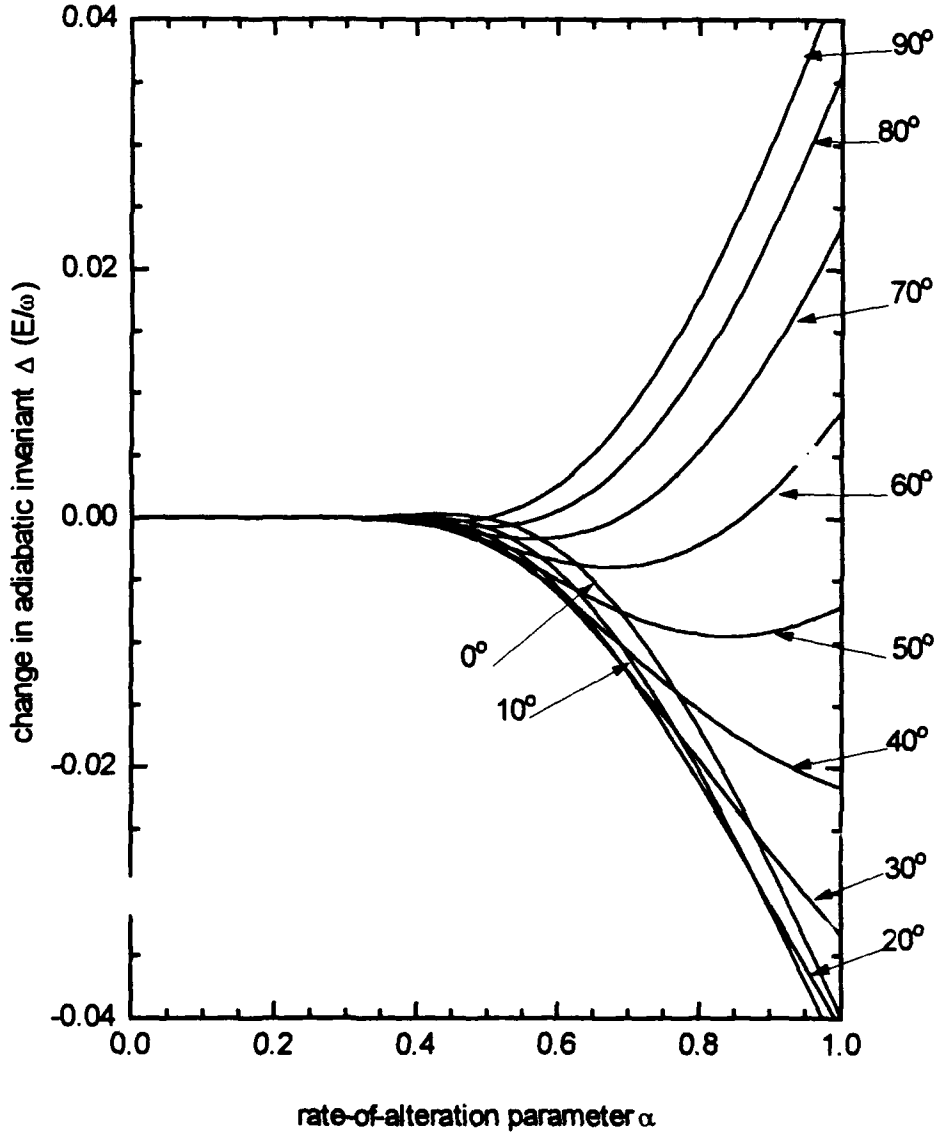


Fig. III.D.3 Numerical simulation results for the *Dilate* model in the intermediate regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.D.1.

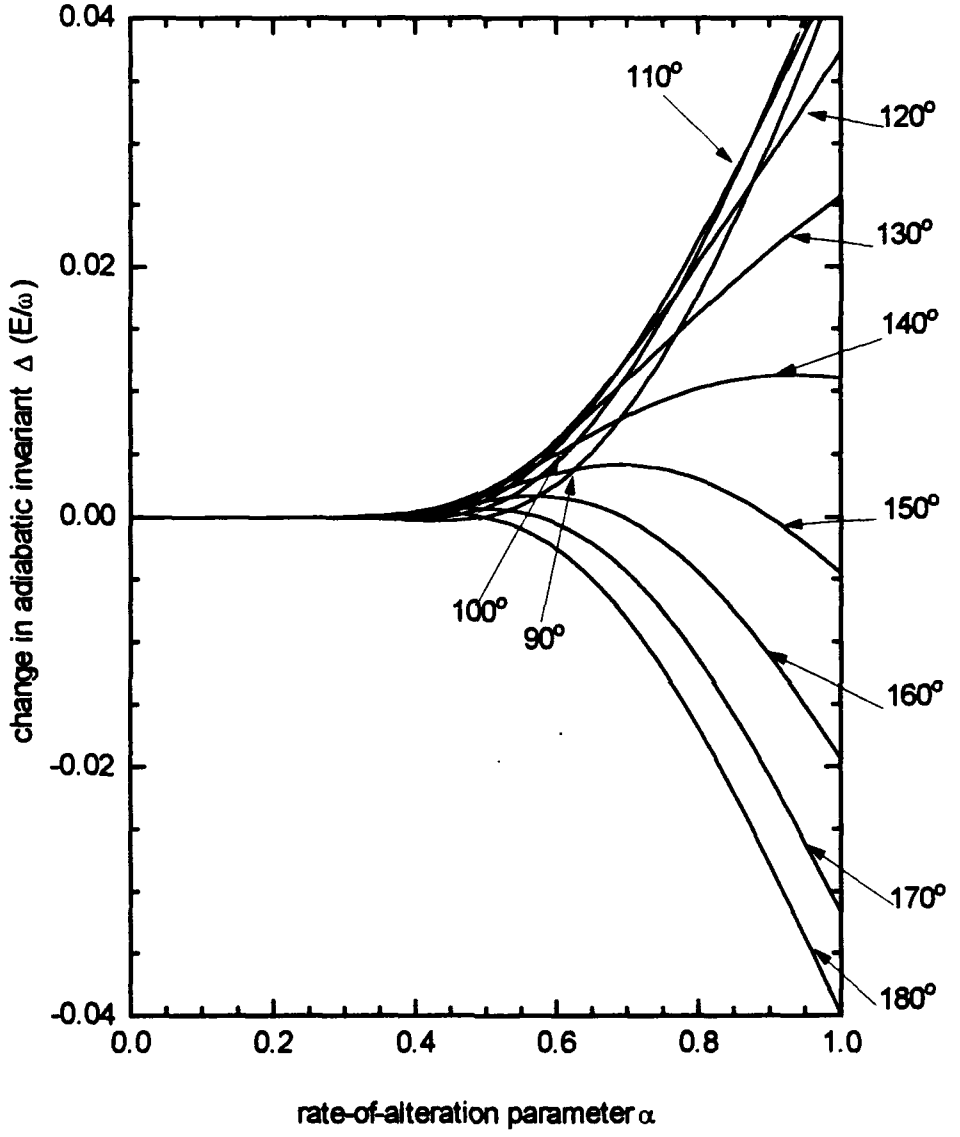


Fig. III.D.4 Numerical simulation results for the *Dilate* model in the intermediate regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.D.1.

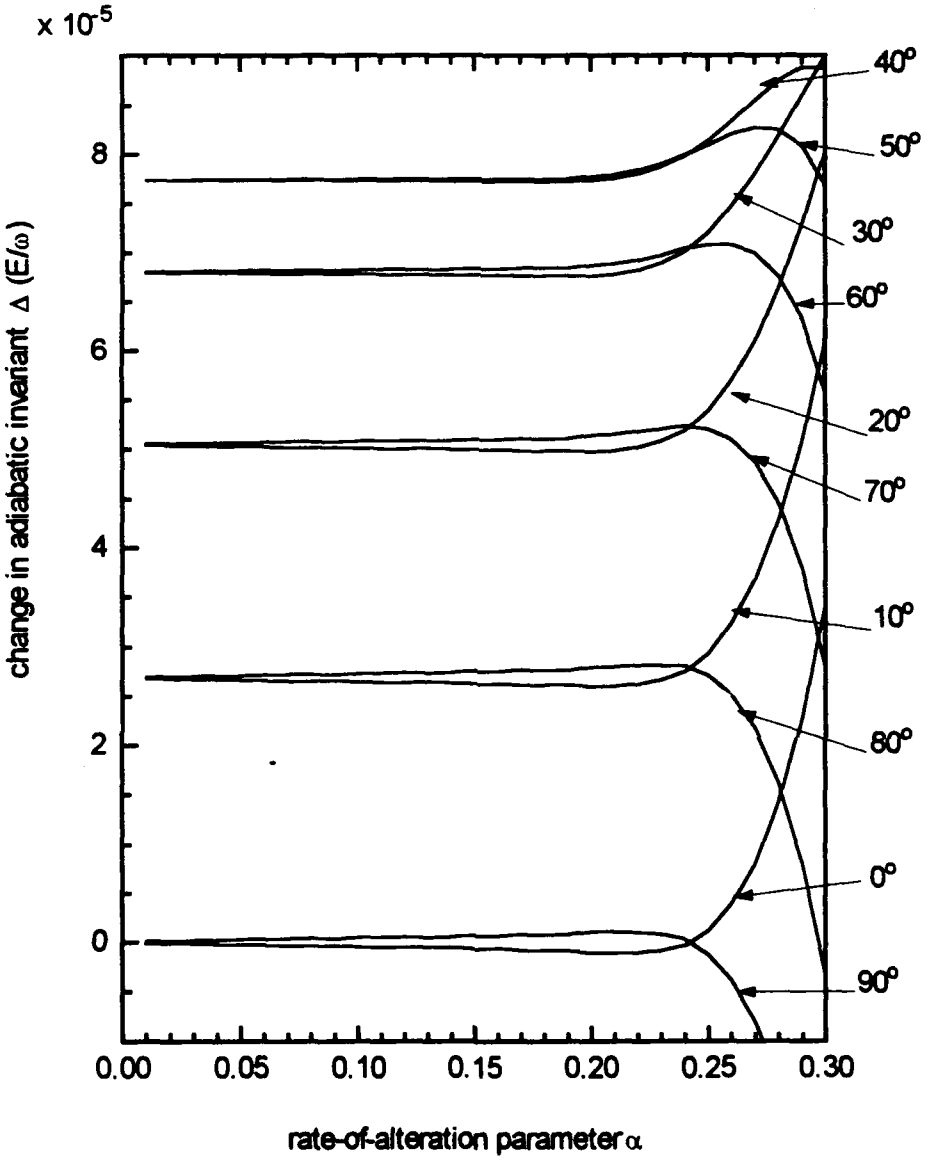


Fig. III.D.5 Numerical simulation results for the *Dilate* model in the near-adiabatic regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.D.1.

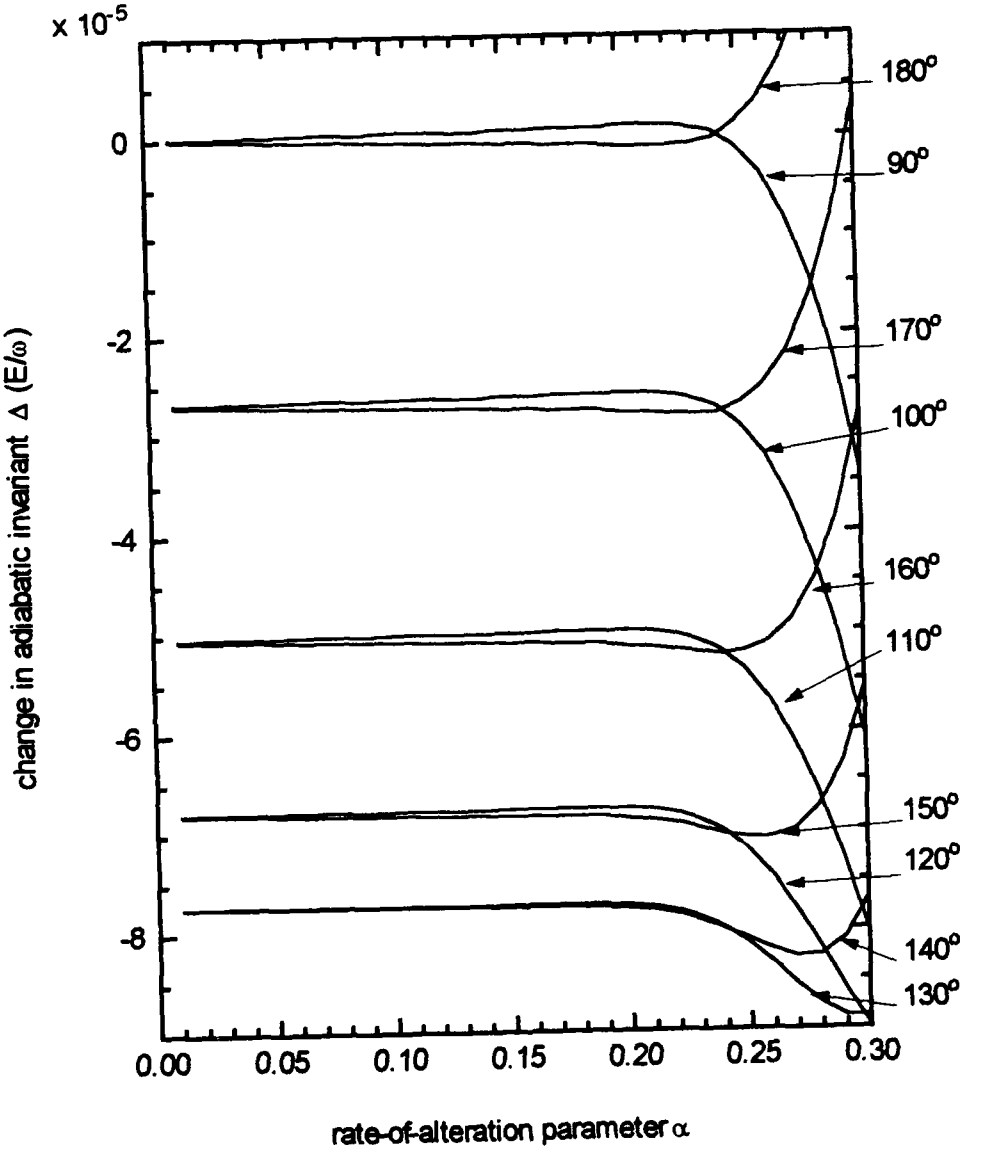


Fig. III.D.6 Numerical simulation results for the *Dilate* model in the near-adiabatic regime. Values of the initial phase are displayed near each curve. The numerical parameters are the same as in Fig. III.D.1.

V. CONCLUSIONS AND FUTURE WORK

We have numerically investigated the change in motion of oscillators subjected to different types of alterations. The rates of alteration were varied over the complete range, from very slow (near-adiabatic regime) to very fast (abrupt regime) compared to the typical period of oscillation. One of the systems (*Rotate*) exhibits interesting behavior in the abrupt regime: The final state can be the rest state for a certain relationship between the values of the rate-of-change parameter and the initial phase. The theory of this system in the abrupt limit fails to describe both this behavior and the asymptotic relationship between the change in amplitude and the rate-of-alteration parameter. The reason for this failure is apparently the singular nature of the alteration; i.e., that the centrifugal force diverges in the limit of an abrupt 360° rotation of the system. Such a singularity is not present in the other two systems (*Translate* and *Dilate*), which behave in the abrupt limit according to the theory, although further investigations are needed (see below).

All three systems show a dramatic suppression of the change in motion as the rate-of-alteration is decreased to values less than roughly the frequency of the motion. This bears out the practical use of adiabatic invariance in situations where the rate of alteration is slow but not very slow, for example, in geometrical ray theory or the bending of microwaves in waveguides. Moreover, our results show that the suppression is sufficiently strong that the rate of alteration need not be slow for the adiabatic invariant to be approximately conserved, and suggest that this may be a general result.

In the near-adiabatic regime, we do not observe the predicted exponentially suppressed breakdown of adiabatic invariance. In the case of *Rotate*, the change in amplitude is linear in the rate-of-alteration parameter. The dependence cannot be

identified in *Translate*, as a result of numerical error. In *Dilate*, the change in adiabatic invariant is discontinuous in the adiabatic limit, which may be in accord with the fact that the alteration function is discontinuous, although this discontinuity is very small (by the amount ε at $|t| = T$, where T is much greater than the period of the motion). In all cases, we observe a remarkable fact not predicted by the theory: The dependence of the change in adiabatic upon the initial phase persists down to the adiabatic limit. This is surprising; it was expected that the sensitivity to initial phase would be lost for very slow alterations as a result of the many cycles of the motion that occur during the alteration. That is, it was expected that the dependence upon initial phase would be "washed out."

There is a substantial amount of future work that can be done as an extension of this thesis. Regarding the near-adiabatic limit, there are the following possibilities: (a) improved numerical accuracy, (b) use of an infinitely smooth alteration function, (c) generalization of the theory to include dependence upon initial phase, (d) integration of the equations of motion, and (e) theoretical determination of the change in final phase. In (a), an initial step could be to reduce the time step in the existing computer programs, allowing them to run for several days. If the results change significantly, then the next step could be to employ a more sophisticated numerical technique in order to increase the accuracy. An interesting aspect here is that the observation of exponential suppression may test the current limits of computational power, similar to simulations probing the famous problem of whether or not the Solar System is stable. In (b), it is possible to construct infinitely smooth functions that vanish outside an interval, although the functions are nonanalytic. Such functions may remove our ε "glitch" problem, or at least move the problem to a more fundamental level. In (c), our results show that the dependence upon initial phase remains in the limit of adiabatic alterations. An important contribution would be to include this in the theory of the weak breakdown of adiabatic invariance, although this

may be a difficult task. In (d), C. Frenzen has pointed out that some of our equations of motion may be integrable. This is especially true for the *Translate* system, whose driving term is independent of the response. In (e), it should not be difficult to theoretically predict the change in final phase for the *Rotate* and *Translate* systems, and to then compare these predictions to the numerical simulations. This may lead to an understanding of the fact that the initial phase dependence persists in *Rotate*, but is lost in *Translate*.

Regarding the abrupt regime, there are the following possibilities for future work: (a) improvement of the theory for *Rotate*, (b) numerical simulations of *Rotate* in the highly abrupt regime, and comparison to theory, and (c) examination of possible 180° increments of ambiguity in the final phase in the *Translate* numerical simulations. In (a), the current approximate theory for *Rotate* is too crude at least in the moderately abrupt regime. More care should be paid to the singularity (i.e., infinite centrifugal force in the abrupt limit). In (b), it may be that the current *Rotate* theory is accurate for extremely abrupt alterations. This should be checked with numerical simulations. In (c), as noted in Sec. III.D, there appear to be ambiguities in the numerical determination of the final phase in *Translate*. This should be checked and corrected if necessary, and the results should then be compared to the abrupt theory.

APPENDIX A ROTATE.C

```
/*************************************************************************
```

This program determines the final state of a one-dimensional oscillator that is rotated by 360 degrees about a perpendicular bisector of the system. The equation of motion for the position of the oscillator is

$$\frac{d^2x}{dt^2} = \left[1 - \Omega(t)\right]x = 0$$

where the rotational angular velocity is given by

$$\begin{gathered} \text{omega}(t < -T) = 0 , \\ \text{omega}(-T <= t <= T) = \text{omega0}*\text{sech}(\text{omega0}*t/2) , \\ \text{omega}(t > T) = 0 . \end{gathered}$$

Without loss of generality, the natural (undriven) angular frequency is chosen to be unity. Also without loss of generality, the initial amplitude is chosen to be unity. For convenience, the initial phase (at t=-T) is constructed to be equal to the phase at t=0 and t=T when there is no rotation (omega=0 for all t). This is ensured by having T/(2*pi) = an integer. The program computes final states for a sequence of values of the peak rotational angular velocity omega0.

```
*************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define         pi      3.14159265358979323846

double omeg0min, omeg0max, omeg0del;
double omega0, T;
double delta, phase0, epsilon;

long int ncycles, ntotal;
long int nsteps;
int type;

void input(void), timestep(void), window(void);
long int maxint(double value);
double omegasq(double t);
```

```
char fname[15];

void main(void)
{
        double x, xold, v, a, t;
        double energy, amp, phase1, phase, phasef, angle;

        long int n, nzero;
        long int m, mtop;

        FILE *fptr, *fopen();

        input();

        fptr = fopen(fname, "w");
        fprintf(fptr, "\n");

        fprintf(fptr, "        omeg0min = %lf\n", omeg0min);
        fprintf(fptr, "        omeg0max = %lf\n", omeg0max);
        fprintf(fptr, "        omeg0del = %lf\n", omeg0del);
        fprintf(fptr, "         epsilon = %le\n", epsilon);
        fprintf(fptr, "          nsteps = %li\n", nsteps);
        fprintf(fptr, "          phase0 = %lf\n", phase0);
        if (type == 1)
        fprintf(fptr, "        rotation = yes\n");
        if (type == 0)
        fprintf(fptr, "        rotation = no\n");
        fprintf(fptr, "\n");
        fprintf(fptr, "The data columns are values of:\n\n");
        fprintf(fptr, "    1)  peak rotational angular velocity\n");
        fprintf(fptr, "    2)  change in amplitude\n");
        fprintf(fptr, "    3)  change in final phase (deg)\n\n");

        printf("\n\n");
        printf("The data columns are values of:\n\n");
        printf("\t1)  data point number\n");
        printf("\t2)  peak rotational angular velocity\n");
        printf("\t3)  change in amplitude\n");
        printf("\t4)  change in final phase (deg)\n\n");

        phase0 = (pi/180.0)*phase0;

        mtop = 1 + maxint(0.9999*(omeg0max - omeg0min)/omeg0del);
        for (m=0; m<=mtop; m++)
```

```
        {
                omega0 = omeg0min + m*omeg0del;

                window();

                timestep();

                x = sin(phase0);
                v = cos(phase0);

                nzero = 0;

                for (n=0; n<=ntotal; n++)
                {
                        t = -T + n*delta;

                        a = (type*omegasq(t) - 1.0)*x;
                        v = v + a*delta;

                        xold = x;
                        x = x + v*delta;
                        if (x*xold < 0.0) nzero++;

                }
                energy = 0.5*(x*x + v*v);
                amp = sqrt(2.0*energy);

                phase1 = 0.5*pi - atan(v/x);
                phase = nzero*pi + phase1;
                phasef = phase0 + 2.0*ncycles*2.0*pi;
                angle = (180.0/pi)*(phase - phasef);

                printf("%li of %li\t\t%lf\t%le\t%lf\n", m+1, mtop+1, omega0, amp-1.0,
angle);
                fprintf(fptr, "    %f    %le    %lf\n", omega0, amp-1.0, angle);
   }

 printf("\nPROGRAM COMPLETED.  The output data file is:  %s\n\n", fname);

 exit(0);

 }


 /************************** end of main program ************************/
```

```
void input(void)

{

char answer;

start:

again:
printf("\n\n");
printf("Enter omeg0min (the minimum peak value omega0\n");
printf("of the rotational angular velocity):  ");
scanf("%lf", &omeg0min);
printf("\n");
if (omeg0min <= 0.0)
{
        printf("\tThe minimum peak value omeg0min\n");
        printf("\tmust be greater than zero.");
        goto again;
}

printf("Enter omeg0max (the maximum peak value omega0\n");
printf("of the rotational angular velocity):  ");
scanf("%lf", &omeg0max);
printf("\n");

printf("Enter omeg0del (the increment of the values of\n");
printf("the peak rotational angular velocity omega0):  ");
scanf("%lf", &omeg0del);
printf("\n");

printf("Enter epsilon (the value of the rotational\n");
printf("angular velocity at the endpoints):  ");
scanf("%lg", &epsilon);
printf("\n");

printf("Enter nsteps [the number of time steps per\n");
printf("smaller cycle (natural response or drive)]:  ");
scanf("%li", &nsteps);
printf("\n");

printf("Enter phase0 (the initial phase in degrees):  ");
scanf("%lf", &phase0);
```

```
printf("\n");

printf("Is the system to be rotated?  [y (yes) for\n");
printf("rotation or n (no) for no rotation]:  ");
scanf("%s", &answer);
if (answer == 'n') type = 0;
else type = 1;
printf("\n");

printf("Enter name of output data file:  ");
scanf("%s", fname);
printf("\n");

printf("The following parameters and name have been chosen:\n\n");
printf("\t\t   omeg0min = %lf\n", omeg0min);
printf("\t\t   omeg0max = %lf\n", omeg0max);
printf("\t\t   omeg0del = %lf\n", omeg0del);
printf("\t\t    epsilon = %lg\n", epsilon);
printf("\t\t     nsteps = %li\n", nsteps);
printf("\t\tphase0 (deg) = %lf\n", phase0);
if (type == 1)
printf("\t\t   rotation = yes\n\n");
if (type == 0)
printf("\t\t   rotation = no\n\n");
printf("\t\t output file = %s\n\n", fname);

question:
printf("Are these the desired values and name of\n");
printf("output data file?  [y (yes) or n (no)]:  ");
scanf("%s", &answer);
printf("\n");
if (answer != 'y'  && answer != 'n') goto question;
if (answer == 'n') goto start;

return;
}


/*************************************************************************/


void window(void)

{
        double prelim;
```

```
        prelim = (2.0/omega0)*log(2.0*omega0/epsilon);
        ncycles = maxint(prelim/(2.0*pi));
        ncycles++;

        T = 2.0*pi*ncycles;

        return;
}

/*************************************************************************/

void timestep(void)

{
        double prelim, maxvalue;

        maxvalue = 1.0;
        if (omega0 > 2.0) maxvalue = 0.5*omega0;

        prelim = (2.0*pi/maxvalue)/nsteps;

        ntotal = maxint(2.0*T/prelim);

        delta = 2.0*T/ntotal;

        return;
}

/*************************************************************************/

double omegasq(double t)

{
        double omega;

        omega = omega0/cosh(0.5*omega0*t);

        return omega*omega;
}
```

```
/*************************************************************************/

long int maxint(double value)

{
        long int intvalue;

        intvalue = (long int)value;

        if (!((double)intvalue <= value) || !((double)(intvalue+1) > value))

        {
                printf("PROGRAM TERMINATED:  Maximum integer function
(maxint)\n");
                printf("                    is not operating correctly.\n\n");
                printf("input value = %lf\t\toutput value = %li\n\n", value, intvalue);
                exit(0);
        }

        return intvalue;
}

/*************************** end of program ***************************/
```

APPENDIX B TRANSLATE.C

```
/*************************************************************************
```

This program determines the final state of a one-dimensional oscillator that is translated a distance of unity along the axis of the system. The equation of motion for the position x of the oscillator in the moving frame of reference is:

$$\frac{dx^2}{dt^2} + x = \text{gforce}(t)$$

where the effective gravitational force is

$$\text{gforce}(t < -T) = 0 ,$$
$$\text{gforce}(-T <= t <= T) = (\text{omega0}/\text{pi})\,\text{sech}(\text{omega0}*t)\tanh(\text{omega0}*t) ,$$
$$\text{gforce}(t > T) = 0 .$$

The velocity of the system is (omega0/pi)sech(omega0*t), and the effective ravitational force is the negative of the time derivative of the velocity (i.e., the negative of the acceleration).

Without loss of generality, the natural (undriven) angular frequency is chosen to be unity. Also without loss of generality, the initial amplitude is chosen to be unity.

For convenience, the initial phase (at t=-T) is constructed to be equal to the phase at t=0 and t=T when there is no translation (gforce=0 for all t). This is ensured by having T/(2*pi) = integer.

The program computes final states for a sequence of values of the peak rate-of-change parameter omega0.

```
*************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define         pi      3.14159265358979323846

double omeg0min, omeg0max, omeg0del;
double omega0, omega0sq, T;
double delta, phase0, epsilon;

long int ncycles, ntotal;
long int nsteps;
int type;
```

```
void input(void), timestep(void), window(void);
long int maxint(double value);
double gforce(double t);

char fname[15];

void main(void)
{
        double x, xold, v, a, t;
        double energy, amp, phase1, phase, phasef, angle;

        long int n, nzero;
        long int m, mtop;

        FILE *fptr, *fopen();

        input();

        fptr = fopen(fname, "w");
        fprintf(fptr, "\n");

        fprintf(fptr, "        omeg0min = %lf\n", omeg0min);
        fprintf(fptr, "        omeg0max = %lf\n", omeg0max);
        fprintf(fptr, "        omeg0del = %lf\n", omeg0del);
        fprintf(fptr, "         epsilon = %le\n", epsilon);
        fprintf(fptr, "          nsteps = %li\n", nsteps);
        fprintf(fptr, "          phase0 = %lf\n", phase0);
        if (type == 1)
        fprintf(fptr, "     translation = yes\n");
        if (type == 0)
        fprintf(fptr, "     translation = no\n");
        fprintf(fptr, "\n");
        fprintf(fptr, "The data columns are values of:\n\n");
        fprintf(fptr, "    1)  peak rate-of-change parameter\n");
        fprintf(fptr, "    2)  change in amplitude\n");
        fprintf(fptr, "    3)  change in final phase (deg)\n\n");

        printf("\n\n");
        printf("The data columns are values of:\n\n");
        printf("\t1)  data point number\n");
        printf("\t2)  peak rate-of-change parameter\n");
        printf("\t3)  change in amplitude\n");
        printf("\t4)  change in final phase (deg)\n\n");
```

```
        phase0 = (pi/180.0)*phase0;

        mtop = 1 + maxint(0.9999*(omeg0max - omeg0min)/omeg0del);
        for (m=0; m<=mtop; m++)
        {
                omega0 = omeg0min + m*omeg0del;
                omega0sq = omega0*omega0;

                window();

                timestep();

                x = sin(phase0);
                v = cos(phase0);

                nzero = 0;

                for (n=0; n<=ntotal; n++)
                {
                        t = -T + n*delta;

                        a = -x + type*gforce(t);
                        v = v + a*delta;

                        xold = x;
                        x = x + v*delta;
                        if (x*xold < 0.0) nzero++;

                }
                energy = 0.5*(x*x + v*v);
                amp = sqrt(2.0*energy);

                phase1 = 0.5*pi - atan(v/x);
                phase = nzero*pi + phase1;
                phasef = phase0 + 2.0*ncycles*2.0*pi;
                angle = (180.0/pi)*(phase - phasef);

                printf("%li of %li\t\t%lf\t%le\t%lf\n", m+1, mtop+1, omega0, amp-1.0,
angle);
                fprintf(fptr, "    %f    %le    %lf\n", omega0, amp-1.0, angle);
   }

 printf("\nPROGRAM COMPLETED. The output data file is: %s\n\n", fname);

 exit(0);
```

```
}

/************************** end of main program ************************/

void input(void)

{

char answer;

start:

again:
printf("\n\n");
printf("Enter omeg0min (the minimum peak value\n");
printf("omega0 of the rate-of-change parameter): ");
scanf("%lf", &omeg0min);
printf("\n");
if (omeg0min <= 0.0)
{
        printf("\tThe minimum peak value omeg0min\n");
        printf("\tmust be greater than zero.");
        goto again;
}

printf("Enter omeg0max (the maximum peak value\n");
printf("omega0 of the rate-of-change parameter): ");
scanf("%lf", &omeg0max);
printf("\n");

printf("Enter omeg0del (the increment of the values of\n");
printf("the peak rate-of-change parameter omega0): ");
scanf("%lf", &omeg0del);
printf("\n");

printf("Enter epsilon (the value of the effective\n");
printf("gravitational force at the endpoints): ");
scanf("%lg", &epsilon);
printf("\n");

printf("Enter nsteps [the number of time steps per\n");
printf("smaller cycle (natural response or drive)]: ");
scanf("%li", &nsteps);
printf("\n");
```

```
printf("Enter phase0 (the initial phase in degrees):  ");
scanf("%lf", &phase0);
printf("\n");

printf("Is the system to be translated?  [y (yes) for\n");
printf("translation or n (no) for no translation]:  ");
scanf("%s", &answer);
if (answer == 'n') type = 0;
else type = 1;
printf("\n");

printf("Enter name of output data file:  ");
scanf("%s", fname);
printf("\n");

printf("The following parameters and name have been chosen:\n\n");
printf("\t\t   omeg0min = %lf\n", omeg0min);
printf("\t\t   omeg0max = %lf\n", omeg0max);
printf("\t\t   omeg0del = %lf\n", omeg0del);
printf("\t\t    epsilon = %lg\n", epsilon);
printf("\t\t     nsteps = %li\n", nsteps);
printf("\t\tphase0 (deg) = %lf\n", phase0);
if (type == 1)
printf("\t\t translation = yes\n\n");
if (type == 0)
printf("\t\t translation = no\n\n");
printf("\t\t output file = %s\n\n", fname);

question:
printf("Are these the desired values and name of\n");
printf("output data file?  [y (yes) or n (no)]:  ");
scanf("%s", &answer);
printf("\n");
if (answer != 'y'  && answer != 'n') goto question;
if (answer == 'n') goto start;

return;
}

/*************************************************************************/

void window(void)

{
```

```
        double prelim;

        prelim = (1.0/omega0)*log(2.0*omega0sq/(pi*epsilon));
        ncycles = maxint(prelim/(2.0*pi));
        ncycles++;

        T = 2.0*pi*ncycles;

        return;
}

/*****************************************************************************/

void timestep(void)

{
        double prelim, maxvalue;

        maxvalue = 1.0;
        if (omega0 > 1.0) maxvalue = omega0;

        prelim = (2.0*pi/maxvalue)/nsteps;

        ntotal = maxint(2.0*T/prelim);

        delta = 2.0*T/ntotal;

        return;
}
/*****************************************************************************/

double gforce(double t)

{
        double arg, value;

        arg = omega0*t;

        value = (omega0sq/pi)*tanh(arg)/cosh(arg);

        return value;
}

/*****************************************************************************/
```

```
long int maxint(double value)

{
        long int intvalue;

        intvalue = (long int)value;

        if (!((double)intvalue <= value) || !((double)(intvalue+1) > value))

        {
                printf("PROGRAM TERMINATED:  Maximum integer function
(maxint)\n");
                printf("                    is not operating correctly.\n\n");
                printf("input value = %lf\t\toutput value = %li\n\n", value, intvalue);
                exit(0);
        }

        return intvalue;
}


/**************************** end of program ****************************/
```

APPENDIX C DILATE.C

```
/*************************************************************************
```

This program determines the final state of a one-dimensional oscillator whose frequency is smoothly changed from an initial value to a final value. The equation of motion for the position x of the oscillator is

$$\frac{d^2x}{dt^2} + \Omega(t)x = 0$$

where the angular frequency is given by

$$\text{omega}(t<-T) = 1 ,$$

$$\text{omega}(-T<=t<=T) = [(\text{omegaf}+1) + (\text{omegaf}-1)*\tanh(\text{omega0}*t)]/2 ,$$

$$\text{omega}(t>T) = \text{omegaf} .$$

Without loss of generality, the initial angular frequency and amplitude are chosen to be unity. The initial value of the adiabatic invariant (energy divided by frequency) is 1/2.

For convenience, the initial phase (at t=-T) is constructed to be equal to the phase at t=0 and t=T when there is no change (omega=1 for all t). This is ensured by having T/(2*pi) = integer.

The program computes final states for a sequence of values of the rate-of-change parameter omega0.

```
*************************************************************************/


#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define     pi      3.14159265358979323846
#define     omegaf  2.0

double omeg0min, omeg0max, omeg0del;
double omega0, T, omega;
double delta, phase0, epsilon;

long int ncycles, ntotal;
```

```
long int nsteps;
int type, typenot;

void input(void), timestep(void), window(void);
long int maxint(double value);
double omegasq(double t);

char fname[15];

void main(void)
{
        double x, xold, v, a, t;
        double energy, amp, adiabat;

        long int n, nzero;
        long int m, mtop;

        FILE *fptr, *fopen();

        input();

        fptr = fopen(fname, "w");
        fprintf(fptr, "\n");

        fprintf(fptr, "        omeg0min = %lf\n", omeg0min);
        fprintf(fptr, "        omeg0max = %lf\n", omeg0max);
        fprintf(fptr, "        omeg0del = %lf\n", omeg0del);
        fprintf(fptr, "         epsilon = %le\n", epsilon);
        fprintf(fptr, "          nsteps = %li\n", nsteps);
        fprintf(fptr, "          phase0 = %lf\n", phase0);
        if (type == 1)
        fprintf(fptr, "          change = yes\n");
        if (type == 0)
        fprintf(fptr, "          change = no\n");
        fprintf(fptr, "\n");
        fprintf(fptr, "The data columns are values of:\n\n");
        fprintf(fptr, "     1) rate-of-change parameter\n");
        fprintf(fptr, "     2) change in adiabatic invariant\n");
        fprintf(fptr, "\n");

        printf("\n\n");
        printf("The data columns are values of:\n\n");
        printf("\t1) data point number\n");
        printf("\t2) rate-of-change parameter\n");
        printf("\t3) change in adiabatic invariant\n");
```

```
        printf("\n");

        phase0 = (pi/180.0)*phase0;

        mtop = 1 + maxint(0.9999*(omeg0max - omeg0min)/omeg0del);
        for (m=0; m<=mtop; m++)
        {
                omega0 = omeg0min + m*omeg0del;

                window();

                timestep();

                x = sin(phase0);
                v = cos(phase0);

                nzero = 0;

                for (n=0; n<=ntotal; n++)
                {
                        t = -T + n*delta;

                        a = -(type*omegasq(t) + typenot)*x;
                        v = v + a*delta;

                        xold = x;
                        x = x + v*delta;
                        if (x*xold < 0.0) nzero++;
                }
                if (type == 0) omega = 1.0;
                if (type == 1) omega = omegaf;

                energy = 0.5*(v*v + omega*omega*x*x);
                amp = sqrt(2.0*energy)/omega;
                adiabat = energy/omega;

                printf("%li of %li\t\t%lf\t%le\n", m+1, mtop+1, omega0, adiabat-0.5);
                fprintf(fptr, "    %f    %le\n", omega0, adiabat-0.5);
        }

printf("\nPROGRAM COMPLETED.  The output data file is:  %s\n\n", fname);

exit(0);

}
```

```
/************************** end of main program ************************/


void input(void)

{

char answer;

start:

again:
printf("\n\n");
printf("Enter omeg0min (the minimum value of\n");
printf("the rate-of-change parameter omega0):  ");
scanf("%lf", &omeg0min);
printf("\n");
if (omeg0min <= 0.0)
{
        printf("\tThe minimum rate-of-change value\n");
        printf("\tomeg0min must be greater than zero.");
        goto again;
}

printf("Enter omeg0max (the maximum value of\n");
printf("the rate-of-change parameter omega0):  ");
scanf("%lf", &omeg0max);
printf("\n");

printf("Enter omeg0del (the increment of the values\n");
printf("of the rate-of-change parameter omega0):  ");
scanf("%lf", &omeg0del);
printf("\n");

printf("Enter epsilon (the deviation of the angular frequency\n");
printf("from the constant values at the endpoints):  ");
scanf("%lg", &epsilon);
printf("\n");

printf("Enter nsteps [the number of time steps per fastest cycle\n");
printf("(during initial, final, or intermediate motion)]:  ");
scanf("%li", &nsteps);
printf("\n");
```

```
printf("Enter phase0 (the initial phase in degrees):  ");
scanf("%lf", &phase0);
printf("\n");

printf("Is the system to be changed?  [y (yes) for\n");
printf("change or n (no) for no change]:  ");
scanf("%s", &answer);
if (answer == 'n') type = 0;
else type = 1;
typenot = 1 - type;
printf("\n");

printf("Enter name of output data file:  ");
scanf("%s", fname);
printf("\n");

printf("The following parameters and name have been chosen:\n\n");
printf("\t\t   omeg0min = %lf\n", omeg0min);
printf("\t\t   omeg0max = %lf\n", omeg0max);
printf("\t\t   omeg0del = %lf\n", omeg0del);
printf("\t\t    epsilon = %lg\n", epsilon);
printf("\t\t     nsteps = %li\n", nsteps);
printf("\t\tphase0 (deg) = %lf\n", phase0);
if (type == 1)
printf("\t\t     change = yes\n\n");
if (type == 0)
printf("\t\t     change = no\n\n");
printf("\t\t output file = %s\n\n", fname);

question:
printf("Are these the desired values and name of\n");
printf("output data file?  [y (yes) or n (no)]:  ");
scanf("%s", &answer);
printf("\n");
if (answer != 'y'  && answer != 'n') goto question;
if (answer == 'n') goto start;

return;
}


/*************************************************************************/
```

```
void window(void)

{
        double prelim;

        prelim = (0.5/omega0)*log(fabs(omegaf*omegaf - 1.0)/epsilon);
        ncycles = maxint(prelim/(2.0*pi));
        ncycles++;

        T = 2.0*pi*ncycles;

        return;
}

/*************************************************************************/


void timestep(void)

{
        double prelim, maxvalue;

        maxvalue = 1.0;
        if (omegaf > 1.0) maxvalue = omegaf;
        if (omega0 > omegaf) maxvalue = omega0;

        prelim = (2.0*pi/maxvalue)/nsteps;

        ntotal = maxint(2.0*T/prelim);

        delta = 2.0*T/ntotal;

        return;
}

/*************************************************************************/


double omegasq(double t)

{
        double omega;
```

```
        omega = 0.5*((omegaf+1.0) + (omegaf-1.0)*tanh(omega0*t));

        return omega*omega;
}

/*************************************************************************/

long int maxint(double value)

{
        long int intvalue;

        intvalue = (long int)value;

        if (!((double)intvalue <= value) || !((double)(intvalue+1) > value))

        {
                printf("PROGRAM TERMINATED:  Maximum integer function
(maxint)\n");
                printf("                    is not operating correctly.\n\n");
                printf("input value = %lf\t\toutput value = %li\n\n", value, intvalue);
                exit(0);
        }

        return intvalue;
}

/*************************** end of program ***************************/
```

LIST OF REFERENCES

Corben, H. C. and P. Stehle, 1960: *Classical Mechanics*, 2nd ed., Wiley, New York, pp. 188-196.

Crawford, Frank S., 1990: "Elementary examples of adiabatic invariance, Am. J. Phys. **58**, 337-344.

Denardo, Bruce and Steven Alkov, 1993: "Multifrequency acoustic resonators with variable nonuniformity," submitted to Am. J. Phys.

Gerald, Curtis F. and Patrick O. Wheatley, 1990: *Applied Numerical Analysis*, Addison-Wesley, Menlo Park.

Gould, Harvey and Jan Tobochnik, 1988: *An Introduction to Computer Simulation Methods*, Addison-Wesley, Menlo Park.

Kubo, Ryogo, 1967: *Statistical Mechanics*, North-Holland, Amsterdam, pp. 57-58, 75-76.

Kvitsinsky, Andrei A. and Seth Putterman, 1990: "Exponentially suppressed transitions in an adiabatically driven quantum system with a discrete spectrum," Phys. Rev. A **42**, 6303-6307.

Landau,L. D., and E. M. Lifshitz, 1976: *Mechanics*, Pergamon, New York, pp. 159-162.

Lenard, A., 1959: "Adiabatic invariants to all orders," Ann. Phys. **6**, 261-276.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center — 2
   Cameron Station
   Alexandria, Virginia 22304-6145

2. Library, Code 052 — 2
   Naval Postgraduate School
   Monterey, California 93943-5000

3. Prof. Bruce Denardo — 5
   Physics Department, PH/De
   Naval Postgraduate School
   Monterey, California 93943-5000

4. Prof. Andres Larraza — 1
   Physics Department, PH/La
   Naval Postgraduate School
   Monterey, California 93943-5000

6. Prof. Anthony Atchley — 1
   Physics Department, PH/Ay
   Naval Postgraduate School
   Monterey, California 93943-5000

7. Prof. William Colson — 1
   Physics Department, PH/Cw
   Naval Postgraduate School
   Monterey, California 93943-5000

8. Prof. Chris Frenzen — 1
   Mathmatics Department, MA/Fr
   Naval Postgraduate School
   Monterey, California 93943-5000

9. C. V. Leigh — 1
   2563 24th Avenue West
   Seattle, Washington 98199-1272